Efficient Algorithms for Separated Continuous Linear Programs: the Multicommodity Flow Problem with Holding Costs and Extensions

Lisa Fleischer* Jay Sethuraman†

June 17, 2004

Abstract

We give an approximation scheme for separated continuous linear programming problems. Such problems arise as fluid relaxations of multiclass queueing networks, and are used to find approximate solutions to complex job shop scheduling problems. In a network with linear flow costs and linear, per-unit-time holding costs, our algorithm finds a drainage of the network, that for given constants $\varepsilon > 0$ and $\delta > 0$ has total cost $(1+\varepsilon)\mathsf{OPT} + \delta$, where OPT is the cost of the minimum cost drainage. The complexity of our algorithm is polynomial in the size of the input network, $\frac{1}{\varepsilon}$, and $\log \frac{1}{\delta}$. The fluid relaxation is a continuous problem. While the problem is known to have a piecewise constant solution, it is not known to have a polynomially-sized solution. We introduce a natural discretization of polynomial size and prove that this discretization produces a solution with low cost. This is the first polynomial time algorithm with a provable approximation guarantee for fluid relaxations.

An appealing feature of the algorithm is that it can explicitly control the time and space complexity of the solution, and also provide a provable bound on its sub-optimality. Given that solutions to fluid relaxations are primarily used to find approximate solutions to discrete scheduling problems, this is a significant advantage in practice, as finding a small-sized near-optimal solution quickly is more valuable than obtaining a more complicated optimal solution with much more effort.

*T. J. Watson Research Center, IBM, Yorktown Heights, NY 10598, USA. Email: lkf@watson.ibm.com. Supported in part through NSF CAREER Award CCR-0049071 and NSF Award EIA-0049084, and in part through IBM Watson Research Center.

†Department of Industrial Engineering and Operations Research, Columbia University, New York, NY 10027, USA, Email: jay@ieor.columbia.edu. Supported in part through NSF CAREER Award DMI-0093981 and IBM Faculty Partnership Award.

1 Introduction

We consider a class of continuous linear programming problems, which arise as a natural model for scheduling and control problems in communication and manufacturing systems. Our main result is a polynomial-time approximation scheme for a basic version of this problem, as well as for several natural extensions. After a formal description of the problem and a brief overview of a motivating application, we discuss related work, and end this section by outlining our contributions.

1.1 Problem description and formulation

We are given a directed network $\mathcal{N} = (V \cup \{s\}, A)$, with commodities $k = 1, \ldots, K$, and a *sink* s; all capacities and costs are non-negative and commodity-dependent. For commodity k, node v has storage capacity $a_k(v)$, per-unit-time linear holding cost $h_k(v)$, and initial supply of $d_k^\circ(v)$; edge e has flow-rate capacity $\mu_k(e)$, and linear flow cost $c_k(e)$. The flow-rate capacity is an upper bound of the flow-rate of commodity k on edge e if e is fully devoted to commodity k. If the use of edge e is divided among several commodities, then the flow-rate capacity for commodity k is $\mu_k(e)$ multiplied by the fraction of edge e alloted to commodity k. Thus, the constraint on edge capacity can be expressed as

$$\sum_k \frac{f_k(e,t)}{\mu_k(e)} \leq 1,$$

where $f_k(e,t)$ is the flow-rate of commodity k on e at time t.

The multiflow problem with holding costs (MHC): We seek a flow (over time) that eventually drains all supplies to the sink s, obeys all the capacity constraints, while minimizing total flow and holding costs.[1] For this problem, it is possible that the optimal solution has exponential complexity: the number of changes in the flow pattern may be exponential in the network size.

We consider two versions of MHC: The *free flow* version, in which commodity k is allowed to travel on any set of paths to reach the sink s; and the *fixed paths* version, where commodity k must travel along a pre-specified path (or set of paths), and the problem is to determine when to continue flow along each arc in the path.

The problem of finding the optimal flow rates $f(\cdot,\cdot)$ for the free-flow version may be formulated as a continuous linear programming problem as described below. We discuss modifications necessary to handle the fixed-paths version in Section 4.2.

[1] While the problem is defined with only one sink, this is without loss of generality: for any $v \in V$ with $h_k(v) = 0$ we create an arc from v to s with infinite capacity, zero cost, (for commodity k) and impose an infinitesimal holding cost on v.

Minimize

$$\sum_k [\sum_{e \in A} c_k(e) \int_0^\infty f_k(e,t)\ dt + \sum_{v \in V} h_k(v) \int_0^\infty d_k(v,t) dt]$$

subject to

$$\forall\ v \in V, t \in R_+, \qquad d_k(v,t) = d_k^\circ(v) - \int_0^t [\sum_{e \in \delta^+(v)} f_k(e,\theta) - \sum_{e \in \delta^-(v)} f_k(e,\theta)] d\theta$$

$$\forall e \in A, t \in R_+, \qquad \sum_k \frac{f_k(e,t)}{\mu_k(e)} \leq 1$$

$$\forall v \in V, t \in R_+, k = 1, 2, \ldots, K, \quad 0 \leq d_k(v,t) \leq a_k(v)$$

$$\forall e \in A, t \in R_+, k = 1, 2, \ldots, K, \quad f_k(e,t) \geq 0$$

In this formulation, $\delta^+(v)$ and $\delta^-(v)$ represent the set of arcs leaving and entering v respectively; and $d_k(v,t)$ represents the storage of commodity k in node v at time t. The first set of constraints conserves flow for each commodity-node pair at each point in time; the second set of constraints restricts the total amount of work an edge can perform at any moment of time; and the final set of constraints enforces the storage capacity for each commodity-node pair at each time.

1.2 Motivation

Consider the production planning problem faced by a manufacturer owning a set of flexible machines. The manufacturer produces a number of products, and a priori estimates of the demand for each product is available. Each product is produced by processing raw material through a fixed sequence of machines ("stages"), requiring varying amounts of processing time at each of the machines in this sequence. Holding costs are used at each stage for each product to capture the opportunity cost of the resources invested. The objective is to produce the required quantities of the various products at minimum cost.

If all of the data are known with certainty, this is simply a job shop scheduling problem with the holding cost objective, which is notoriously difficult to solve exactly. To find reasonable solutions to this scheduling problem, it is natural to consider "tractable" relaxations. One such relaxation is obtained by treating jobs as continuously divisible entities ("fluid"),

and by allowing machines to split their processing capacity among multiple products. The relationship to the fixed-paths version of the multiflow problem with holding costs is now clear: each product represents a commodity, the sequence of machines associated with that product specifies the route through the network of this commodity, etc. This relaxation is called the fluid relaxation associated with the scheduling problem. Two natural issues arise: solving the fluid relxation, and deriving a good schedule for the original problem based on an optimal fluid solution. Our results on the multiflow problem with holding costs imply that we can efficiently find a near-optimal solution to the fluid relaxation associated with this scheduling problem.

A variety of scheduling and control problems arising in communication and manufacturing systems can be modeled using the notion of a multiclass queueing network. Multiclass queueing networks serve as useful models for problems in which several types of *activities* compete for a limited number of shared *resources* [13, 15, 22]. They generalize conventional job-shop problems in two ways: jobs arrive *over time*, and each job has a *random* processing time at each stage. The optimal control problem in a multiclass queueing network is to find an optimal allocation of the available resources to activities over time. Recognizing the importance and the inherent intractability of this problem, the research community has focused its attention, for the most part, on developing tractable approximations [4, 11, 12, 23, 24, 28, 30, 37]; one of the most effective ways to address this optimal control problem is via *fluid relaxations*, which are deterministic, continuous approximations to these stochastic, discrete networks. To get a fluid relaxation of the multiclass queueing network, we replace *discrete* jobs moving *stochastically* through the network by a *continuous*, *deterministic* fluid flow. In addition, we allow a resource to be "shared" among multiple activities simultaneously. Again, two issues arise: first, the tractability of the fluid relaxation itself, and second, the use of a fluid solution to derive an implementable solution for the original control problem. Recent research that has focused on the second issue includes finding near-optimal schedules for deterministic job shop problems with the makespan and holding cost objectives [9, 10], asymptotically optimal schedules for stochastic job shops with the makespan objective [14], and asymptotically optimal schedules for multiclass queueing networks [6, 29]. All of these results rely on the solution to associated fluid relaxations. While the fluid relaxation for the makespan objective is solvable in closed form, the case of linear holding costs is significantly more difficult. This latter problem is an instance of the class of continuous linear programming problems considered here.

1.3 Previous work and related problems

Continuous linear programs were introduced by Bellman [7, 8], who studied a linear optimal control problem in production planning. In spite of a tremendous amount of effort, general continuous linear programs remain difficult to solve [2]. Our interest in these problems is due to their ability to model a variety of dynamic resource allocation problems; fortunately, the problems of interest in these applications have a special structure [2, 32], which we exploit to provide efficient solutions.

Fluid relaxations are a specially structured class of continuous linear programs called *state constrained separated continuous linear programs* (SCSCLP). In the absence of upper bounds on storage, these are called *separated continuous linear programs* (SCLP) [2]. The flow-rate functions on the arcs are the "control" variables, and the storage at the nodes are the "state" variables; the term "separated" refers to the absence of state feedback. SCLPs were first introduced by Anderson [1] as a continuous model for job shop scheduling. Anderson, Nash, and Perold [3] characterized the extreme point solutions to SCLP. In addition, for problems with linear data, they showed the existence of an optimal solution in which the flow-rate functions are piecewise constant (hence, piecewise linear node-storages) with a finite number of pieces. The complexity of SCLP is still unresolved; in fact, the *size* of the optimal solution may be exponential in the input size. Indeed, for the free-flow version of the problem, Rote [36] developed a family of examples that suggests exponential growth in the number of pieces in an optimal solution. The optimal solutions in these examples sends flow along (increasingly longer) cycles to reduce holding costs.

In a series of papers [32, 33, 34, 35], Pullan carried out an extensive study of SCLPs and variants; he proposed an elegant dual for this problem, established strong duality, and designed a class of convergent algorithms, based on time-discretization. Pullan's algorithm starts with a guess of the breakpoints in the optimal solution. With respect to this fixed set of breakpoints, the problem can be solved as a linear program. To compute a lower bound, another linear program with twice as many breakpoints is constructed, with a slightly modified cost function; the cost function is modified in such a way that every feasible solution to its dual can be used to construct a feasible solution to the dual of the original continuous linear program with identical cost. Thus, by solving these two (ordinary) linear programs, one can estimate the duality gap. If the gap is not small enough, the number of breakpoints is doubled, with a new breakpoint added at the midpoints of the original breakpoints. As one can see, a naive implementation of this algorithm becomes impractical soon; to overcome

[2] There are two equivalent SCLP formulations in the literature: One is as a linear optimal control problem, and the other is as a flow problem over time, see [2, pp. 135-136]. We use the flow formulation here.

this difficulty, variants have been developed in which redundant breakpoints are identified and removed every once in a while [31], leading to the so-called adaptive discretization algorithms. Luo and Bertsimas [27] introduced SCSCLP, established strong duality, and proposed a convergent class of algorithms for this problem. Their algorithm is also based on time discretization, removes redundant breakpoints, but solves quadratic programs in intermediate steps. Recently, Weiss has announced a simplex algorithm for separated continuous linear programming [38]. All of these algorithms guarantee convergence, but provide neither a bound on the number of iterations needed, nor a bound on the number of breakpoints in the solution computed.

In the special case when all holding costs are equal, the problem is solved by a flow that minimizes the total supply left in the network at every moment in time. Optimal solutions for this problem (called an *earliest arrival transshipment*) along with polynomial time algorithms to compute it are described in [21, 16]. A more complicated problem that is not known to have a polynomial-sized solution is the problem of minimizing the total time flow takes to reach the sink from a specified source when it takes flow time to travel from the tail of an edge to the head of an edge. This is the earliest arrival flow problem with transit times. For this problem, Hoppe and Tardos described a fully polynomial approximation scheme [26]. When in addition there are multiple sources, a fully polynomial approximation scheme is described in [17].

One key difference between universally quickest flows (with uniform holding costs and with or without travel times) and MHC (with general holding costs) is that an optimal solution to MHC may require sending flow on non-simple paths, while optimal solutions to universally quickest flows never require this.

The MHC problem on a line – a tandem network – for the special case when holding costs are nondecreasing as they approach the sink s is solvable in polynomial time [5].

1.4 Our Contribution

Our main contribution is the first provably efficient algorithm for approximately solving MHC: our algorithm works for both the free-flow and the fixed-paths versions. Given constants $\varepsilon > 0$ and $\delta > 0$, we find a solution with total cost at most $(1+\varepsilon)\mathsf{OPT} + \delta$, where OPT is the cost of the minimum cost drainage. The complexity of our algorithm is polynomial in the size of the input network, $\frac{1}{\varepsilon}$, and $\log \frac{1}{\delta}$. This algorithm is described in Section 4.

Our main result extends to generalizations of MHC that include piecewise-constant data, convex holding costs, and arbitrary additional convex constraints. These extensions are discussed in Section 5.

Our algorithm also uses time discretization, but, in contrast to previous approaches for

MHC and SCLP, our algorithm works with a *fixed* time partition. A *fixed* time partition is used previously in the approximation scheme to minimize total time the flow spends in the network when there are transit times and multiple sources [17]. We prove that the optimal instantaneous holding cost function is a convex, decreasing function, and use this to devise strong lower bounds for the problem based on the time partition. We use a time expanded network with side constraints, with network copies representing geometrically increasing units of time. Our algorithm finds a flow with constant flow rates within each time interval in the partition. This is in contrast to prior discretization-based algorithms [32, 27] which adaptively refine the discretization, and are unable to bound the number of breakpoints in the computed solution. Our approximation scheme provides a systematic way to control the *solution complexity*: if a solution with a small number of breakpoints is desired, our scheme could be adapted by suitably choosing ϵ and δ.

In addition to providing the desired solution, our algorithm also provides a bound on the sub-optimality of the given solution. In particular, our algorithm may be used in an adaptive setting: given a solution produced by our algorithm, the contribution towards improving the approximation guarantee of individual breakpoints can be assessed, and then removed if deemed small enough. Alternatively, the algorithm can start with a coarse discretization and then the returned solution and bound will suggest which intervals would be best to refine in order to improve the value of the solution.

In summary, we believe the significance of our work is a simple and efficient method to find provably good solutions of small size to separated continuous linear programs. From a practical point of view, finding a near-optimal solution is more useful than finding an optimal solution for the following two reasons:

- Fluid relaxations are primarily used to find good approximate solutions to discrete scheduling problems, so the advantage of having an optimal solution over a near-optimal one is not clear. In fact, the opposite is true: finding a near-optimal solution quickly is more valuable than finding an exact solution laboriously.

- We do not have an apriori bound on the size of an optimal solution. If an optimal solution does not have small size, finding a near-optimal solution of small size is clearly more valuable. As we mentioned earlier, a family of examples due to Rote [36] suggests that the size of an optimal solution may be exponential in the input size.

2 Preliminaries

2.1 Input form and size

Our network has $n = |V|$ vertices, $m = |E|$ arcs, and K commodities. While the control problem in fluid networks is defined for arbitrary input, we assume that we are handling numerical input specified as the ratio of two integers, the maximum of which is bounded by U. Thus the size of the input to the problem can be expressed as a polynomial in terms of n, m, and $\log U$. We denote this polynomial by $p(n, m, K, \log U)$.

Without loss of generality, we assume that the capacity function μ is integral. This can be done by multiplying capacities and demands by the least common multiple of capacity denominators, and dividing the costs by the identical number. The solution to the resulting problem has the same cost as the original, and can be transformed into a solution to the original problem simply by dividing the flow rate at each moment of time by the same scaling factor.

Our algorithm requires a bound on the optimal *time-horizon* — the amount of time required by the optimal flow to empty the network. A trivial upper bound on the optimal time horizon, if finite, is simply $\sum_k \sum_{v \in V} d_k^\circ(v)$, since at worst, the network drains flow at a rate equal to the minimum capacity, which is at least one if the problem is feasible. Thus, for the rest of the paper, we assume $T = \sum_k \sum_{v \in V} d_k^\circ(v) \leq nK|U|$.

2.2 Notation and Definitions

We use $f(t)$ to denote control f at time t. We use $f(e)$ to denote the K-component vector of functions of time that describe the control of each commodity on arc e. We use $f(e, t)$ to denote the vector of specific commodity flow values on e at time t. An optimal control is denoted f^*.

Control f and initial storage d° induce a vector of commodity-per-vertex storage functions, denoted d_f, or d when f is clear from context. We use $d_k(v, t)$ to denote the amount of commodity k stored at v at time t. We use $d(v, t)$ to denote K-component storage vector at v at time t. We use $d(t)$ to denote d vector evaluated at time t. The storage function vector of an optimal control f^* is denoted d^*.

We abbreviate the objective function $\sum_k [\ \sum_{e \in A} c_k(e) \int_0^\infty f_k(e, t)\ dt\ +\ \sum_{v \in V} h_k(v) \int_0^\infty d_k(v, t)\ dt\ ]$ as $\int_0^T [c^\mathsf{T} f(t) + h^\mathsf{T} d(t)]\ dt$, for the appropriate upper bound T, and refer to the instantaneous value at t as $c^\mathsf{T} f(t) + h^\mathsf{T} d(t)$.

Given an interval (of time) $[a, b)$ or $[a, b]$, the *length* of the interval is $b - a$.

3 Structure and Use of the Discretization

A key tool in our algorithm is a *non-uniform* time expanded network. Section 3.1 describes the structure and properties of this network. Section 3.2 describes some structure of the optimal solution. Section 3.3 combines the content of these two previous sections to develop a new lower bound for the optimal control problem that we use to prove approximate optimality of our algorithm.

3.1 Time-expanded networks

We can compute a feasible, but not, in general, optimal control by using a uniform time-expanded network. A *time-expanded* network of $\mathcal{N} = (V, A)$ with time horizon T is denoted $\mathcal{N}^T$ and contains a copy of $\mathcal{N}$ for every time interval in $[0, T)$ of the form $[\theta, \theta + 1)$ for $\theta = 0, 1, \ldots, T - 1$. The copy for interval $[\theta, \theta + 1)$ is denoted V_θ. The copies of vertex v and arc e in V_θ are denoted v_θ and e_θ, respectively. The flow capacity restrictions on $e \in A$ are interpreted as flow capacity restrictions for e_θ for each $\theta = 0, \ldots, T - 1$. In addition, if storage is permitted at v, then there is a *holdover arc* from v_θ to $v_{\theta+1}$ of capacity $a_k(v)$ for each commodity $k = 1, \ldots, K$, for all $\theta = 0, \ldots, T - 1$. Finally, there are holdover arcs $(s_\theta, s_{\theta+1})$ of infinite capacity for all $\theta = 0, \ldots, T - 1$.

A standard flow in the time-expanded network $\mathcal{N}^T$ corresponds to the control f obtained by interpreting the flow on arc e_θ as the flow rate on e in the interval $[\theta, \theta+1)$ and interpreting the flow on arc $(v_\theta, v_{\theta+1})$ as the storage level at v at time $\theta + 1$. Since the obtained flow rates are constant on unit intervals, this completely specifies f. Similarly, any control f corresponds to a flow x in $\mathcal{N}^T$: x is obtained by averaging f on unit intervals. See Figure 1 for an illustration of this correspondence.

We now discuss how to assign costs to arcs in $\mathcal{N}^T$ so that the cost of a flow in $\mathcal{N}^T$ is the same as the cost of the corresponding control. This corresponding control is constant over unit intervals which implies that the storage function d is linear over unit intervals. We begin by examining the cost behavior for a single interval. Since d is linear in this interval, the holding cost for commodity k at v in interval $[\theta, \theta+1)$ is $\int_\theta^{\theta+1} h_k(v)^\mathsf{T} d_k(v, t)\, dt = \frac{1}{2}[h_k(v)^\mathsf{T} d_k(v, \theta) + h_k(v)^\mathsf{T} d_k(v, \theta+1)]$. As the flow of commodity k on the holdover arc entering v_θ corresponds to $d_k(v, \theta)$, we assign a cost of $\frac{1}{2}h_k$ to flow of commodity k on this arc to account for holding cost of flow that starts the interval at v. Similarly, since the flow of commodity k on the holdover arc leaving v_θ corresponds to $d_k(v, \theta + 1)$, we assign a cost of $\frac{1}{2}h_k(v)$ to this arc to account for holding cost of flow that ends the interval at v. Flow that stays at v incurs both costs. We do this for all intervals.

Putting the intervals together into one network, we create the *time-expanded network with*

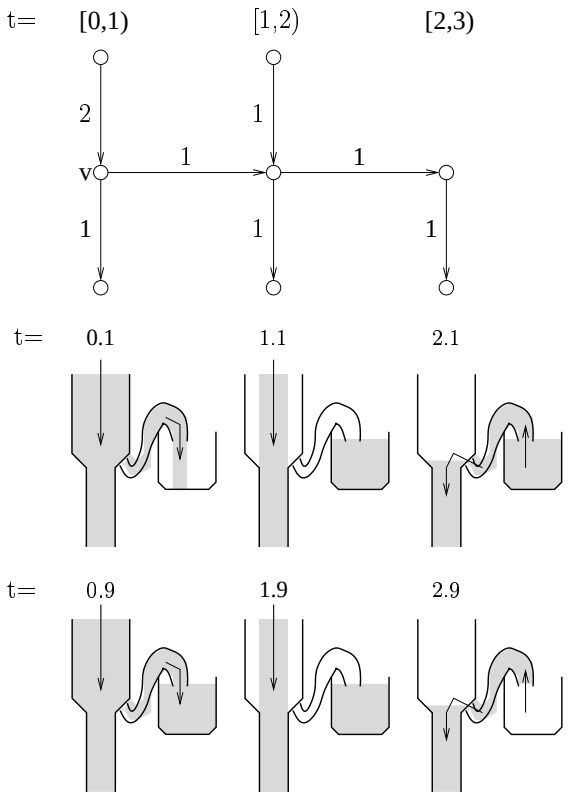

Figure 1: The flow in a time-expansion of a network fragment consisting of 3 nodes and two arcs: one with capacity 2 and another with capacity 1. The pipes below depict the behavior of the corresponding flow over time at times 0.1, 0.9, 1.1, 1.9, 2.1, 2.9. In the interval $[0, 1)$, the reserve bucket is gradually filled by excess flow arriving at node v. In interval $[1, 2)$ this reserve flow sits at node v since the arc leaving v is full. In interval $[2, 3)$ the reserve bucket is gradually emptied.

costs. This is a modification of a time expanded network $\mathcal{N}^T$ created by adding a new vertex v'_θ for each vertex v_θ in $\mathcal{N}^T$. See Figure 2. The arc set of $\mathcal{N}^T$ is modified by replacing each holdover arc $(v_\theta, v_{\theta+1})$ with two arcs $(v_\theta, v'_{\theta+1})$ and $(v'_{\theta+1}, v_{\theta+1})$. The new arcs each have the capacity of the old arc, and cost $h_k(v)/2$ for commodity k. For each vertex $v \in V$, the arc (v'_0, v_0) is introduced with capacity $d^\circ_k(v)$ and cost $h_k(v)/2$ for commodity k, $k = 1, \ldots, K$. Arc e_θ has cost $c_k(e)$ for commodity k. For $\mathcal{N}^T$, denote this modified network with costs as $\mathcal{N}^T_c$. Note that, aside from the first vertex v'_0, the set of added vertices are unnecessary for accurate computation so far. We add them to separate the contribution to the holding cost of each interval. This becomes more important in Sections 3.1.1 and 3.1.2 where we modify $\mathcal{N}^T_c$ further.

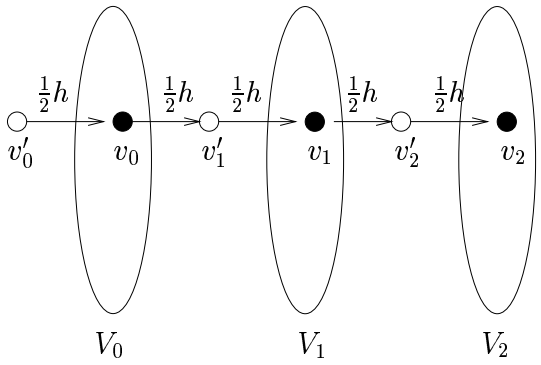

Figure 2: A schematic sketch of the modified time-expanded network with holding costs. The flow on arc (v_{i-1}, v'_i) (or arc (v'_i, v_i)) denotes flow stored at v at time i.

Theorem 3.1 *A flow x in $\mathcal{N}_c^T$ that sends, for all $v \in V$, $k = 1, \ldots, K$, $d_k^\circ(v)$ units of flow from v'_0 to s_T corresponds to a control f in $\mathcal{N}$ with the same cost.*

Proof: Given x, let f be the piecewise constant flow obtained by interpreting $x_k(e_\theta)$ as the flow rate of commodity k on e in $[\theta, \theta+1)$ for all $k \in \{1, \ldots, K\}$, $e \in A$. Since f is constant on unit intervals, the rate of drainage from $v \in V$ in $[\theta, \theta+1)$ is constant on this interval. Thus the holding cost at v in this interval is $\sum_k [\frac{1}{2} h_k(v) | d_k(v, \theta) - d_k(v, \theta+1)| + h_k(v) \min\{d_k(v, \theta), d_k(v, \theta+1)\}]$. For $0 \le \theta \le T-2$, this is captured by x as the cost of the flow of commodity k on (v'_θ, v_θ) plus the cost of the flow of commodity k on $(v_\theta, v'_{\theta+1})$. For $\theta = T-1$, this is the cost of flow of commodity k on (v'_{T-1}, v_{T-1}), since $d_k(v, T) = 0$ for all k. The flow cost on this interval is simply the sum of the flow costs on arcs in V_θ. □

Corollary 3.2 *If f^* sends flow at a rate that is constant on unit intervals, then a minimum cost flow in $\mathcal{N}_c^T$ yields a minimum cost control.*

Unfortunately, we cannot use Corollary 3.2 to obtain an optimal control f^* in general since there is no guarantee that f^* is constant on unit intervals. If f^* sends a lot of flow from node v at the beginning of an interval, and very little at the end, then the holding cost at v during the interval will be significantly lower with f^* than with the flow obtained by averaging f^* over the interval. For example, consider a buffer with holding cost 1 and one unit of flow, and an arc leaving the buffer with capacity ten. If the flow is sent at maximum capacity from the start, then the holding cost is $\int_0^{1/10} (1 - 10x) dx = 1/20$. If the flow is kept in the buffer as long as possible and sent at maximum capacity at the end of the unit interval, the holding cost is $9/10 + \int_0^{1/10} (1 - 10x) dx = 19/20$. The average of either of these flows is the flow that sends flow at rate $1/10$ of capacity throughout the unit interval, and this has

holding cost $\int_0^1 (1-x)dx = 1/2$. There are symmetric cost disparities for the case of flow that is entering the buffer. We will address this difficulty by refining the intervals of discretization selectively. A key structural property that allows for this is given in Lemma 3.4.

Even if f^* is constant over unit intervals, the algorithm implied by computing a minimum cost flow in $\mathcal{N}_c^T$ is pseudopolynomial: its complexity depends polynomially on $|U|$, and hence is exponential in the size of the input parameter $\log |U|$. Thus, to obtain a polynomial algorithm, it is necessary to work with smaller networks.

3.1.1 Condensed Time-Expanded Networks

Instead of using V_θ to represent one unit of time, we can instead use V_θ to represent a time interval of length Δ. In an interval of length Δ, an arc can carry Δ times its capacity. Thus, the capacity of commodity k on each arc in V_θ is multiplied by Δ. The cost of carrying such flow remains as before, so the cost on this arc remains the per-unit-commodity flow cost. Flow held at v over an interval of length Δ has holding cost of Δ times the per-unit-time holding cost of that flow. Thus the cost on arcs entering and leaving V_θ are multiplied by Δ. (That is, cost of commodity k on (v'_θ, v_θ) and $(v_\theta, v'_{\theta+1})$ is multiplied by Δ for all $v \in V$, $k = 1, \ldots, K$, as in Figure 3.) However, the bound on the amount of flow that can be held at v does not change, so the capacity of these arcs remain as before.

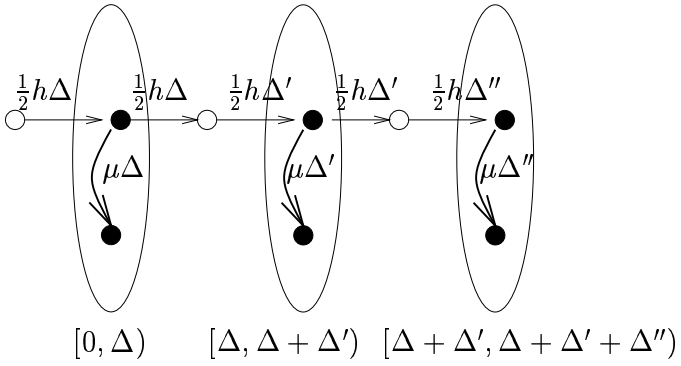

Figure 3: A schematic sketch of a nonuniform, condensed time-expanded network with breakpoints in the set $\{0, \Delta, \Delta + \Delta', \Delta + \Delta' + \Delta''\}$. Capacities of network arcs are multiplied by the interval length. Holding costs into and out of network representing fixed interval are also multiplied by interval length.

Flow in this *condensed* time-expanded network corresponds to a control by dividing the flow on arc e_θ by Δ: If V_θ corresponds to the interval $[a, a + \Delta)$ then the control sends flow onto e at rate $x(e_\theta)/\Delta$ for this entire interval. The storage level of commodity k at v at time

$a + \alpha\Delta$ for $\alpha \in [0,1]$ is $(1-\alpha)\ x_k(v'_\theta, v_\theta) + \alpha\ x_k(v_\theta, v'_{\theta+1})$.

The effect on the corresponding control of condensing the interval $[a, a+\Delta)$ in the time-expanded network to just one copy of $\mathcal{N}$ is to average the control over a longer interval – an interval of length Δ. Averaging flow over an interval has no affect on the flow costs, as the flow costs are per-unit-flow entering an arc. However, this increases the holding costs, since they are per-unit-time. Thus, the cost of a minimum cost flow and corresponding control is higher in a condensed time expanded network with intervals of length 2Δ than it in the comparable condensed network with intervals of length Δ.

3.1.2 Nonuniform Time-Expanded Networks

The condensed time-expanded network defined in the previous section may be further generalized so that each copy of the network can represent an interval of time of a different length. We will represent such interval lengths by a set of breakpoints. A set $\mathcal{B}$ of consecutive breakpoints in $[0,T]$ with $0, T \in \mathcal{B}$ defines a set of disjoint intervals that covers $[0,T)$. The corresponding time-expanded network, denoted $\mathcal{N}^T_{c,\mathcal{B}}$, is the time expanded network that contains a copy of V for every such interval. The proof of the following theorem is similar to the proof of Theorem 3.1.

Theorem 3.3 *A flow x in $\mathcal{N}^T_{c,\mathcal{B}}$ that sends, for all $v \in V$, $k = 1, \ldots, K$, $d^\circ_k(v)$ units of flow from v'_0 to s_T corresponds to a control f in $\mathcal{N}$ with the same cost.*

A simple, 2-level nonuniform time-expanded network is introduced in [16] to compute a quickest transshipment. This framework is extended in [17, 18] to handle nonzero transit times.

3.2 Structure of an Optimal Solution

It is easy to see that an optimal solution may send flow on non-simple paths. In particular, it may be better to send excess supplies to a vertex with cheap holding costs while waiting for sufficient capacity to the sink. However, as the following lemma implies, the total holding cost accrued (over all nodes) in a unit interval decreases with time.

Lemma 3.4 *$h^\mathsf{T} d^*(t)$ is a convex, decreasing function of t.*

Proof: If $h^\mathsf{T} d^*$ is not convex, then there is a lower tangent l to $h^\mathsf{T} d^*$ with discontinuous intersection with $h^\mathsf{T} d^*$. Let $0 < t_1 < t_3 < t_2$ be such that $h^\mathsf{T} d^*(t_1)$ and $h^\mathsf{T} d^*(t_2)$ are on l, $h^\mathsf{T} d^*(t_3)$ is not on l, and for all $t_1 \le t \le t_2$, $h^\mathsf{T} d^*(t)$ is on or above l. Modify f^* on the interval $[t_1, t_2)$ by replacing $f^*(e,t)$ with the average flow rate $\frac{1}{t_2 - t_1} \int_{t_1}^{t_2} f^*(e,t)\ dt$ for all $e \in A$ and

all $t \in [t_1, t_2)$. Call the new control $\bar{f}$. Since f^* obeys capacity constraints, so does $\bar{f}$. The flow costs of f^* and $\bar{f}$ are the same, since $\bar{f}$ uses the same routing as f^* does. However, the holding costs are cheaper, since now the storage at v changes at a uniform rate over $[t_1, t_2]$ from $d^*(t_1) = d_{\bar{f}}(t_1)$ to $d^*(t_2) = d_{\bar{f}}(t_2)$, and thus, by choice of t_1 and t_2, the average amount of flow stored per time over the interval has decreased. This contradicts the optimality of f^*. Hence $h^\mathsf{T} d^*$ is convex.

Since $h^\mathsf{T} d^*(0) = h^\mathsf{T} d^\circ > 0$, $h^\mathsf{T} d^*(T) = 0$, and $h^\mathsf{T} d^*$ is convex and nonnegative, we have that $h^\mathsf{T} d^*$ is also decreasing. □

Notice that the above proof extends to show that $h^\mathsf{T} d^*(t)$ is convex decreasing even when f^* is restricted to send flow of commodity k along a prespecified path.

This proof extends trivially to the case of a control computed via a minimum cost flow in $\mathcal{N}^T_{c,\mathcal{B}}$. We summarize this in the following lemma.

Lemma 3.5 *The piecewise constant control f obtained from a minimum cost flow in $\mathcal{N}^T_{c,\mathcal{B}}$ yields a storage function vector $d(t)$ so that $h^\mathsf{T} d(t)$ is a convex, decreasing function of t.*

Proof: A minimum cost flow in $\mathcal{N}^T_{c,\mathcal{B}}$ yields a piecewise constant control f with breakpoints in $\mathcal{B}$. This results in a piecewise linear storage function vector $d(t)$, also with breakpoints in $\mathcal{B}$. If this is not convex, there is a lower tangent l to $h^\mathsf{T} d^*$ with discontinuous intersection with $h^\mathsf{T} d^*$. Let $0 < t_1 < t_3 < t_2$ be such that $h^\mathsf{T} d^*(t_1)$ and $h^\mathsf{T} d^*(t_2)$ are on l, $h^\mathsf{T} d^*(t_3)$ is not on l, and for all $t_1 \le t \le t_2$, $h^\mathsf{T} d^*(t)$ is on or above l. Since $d(t)$ has its breakpoints in $\mathcal{B}$, we may take t_1 and t_2 to be in $\mathcal{B}$. By the same arguments used in the proof of Lemma 3.4, we can average the flow from t_1 to t_2 and produce a new flow with smaller holding cost. This new flow is also piecewise constant with breakpoints in $\mathcal{B}$, and thus is obtainable as a flow in $\mathcal{N}^T_{c,\mathcal{B}}$. This contradicts the fact that we start with a minimum cost flow in $\mathcal{N}^T_{c,\mathcal{B}}$. □

3.3 Strong Lower Bounds

Theorem 3.1 describes how to obtain upper bounds on the cost of a minimum cost control. To obtain a lower bound, we combine ideas of sections 3.1 and 3.2. We give two related lower bounds. The weaker one requires only that the holding costs be nondecreasing; while the second is stronger and uses the convexity of the holding cost over time.

A geometric interpretation of the holding cost portion of these two lower bounds is presented in Figure 4. The top line in each graph is the instantaneous holding cost of the optimal solution as a function of time. Thus the total holding cost is the area under this line. Given a set of breakpoints $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$ with $0 < b_1 < \cdots < b_{|\mathcal{B}|} = T$, the bottom function in the graph on the left is the decreasing step function,

$$l_{\mathcal{B}}(t) := h^\mathsf{T} d^*(b_\theta), \quad \text{for all } t \in (b_{\theta-1}, b_\theta], \text{ and for all } \theta \in \{1, \ldots |\mathcal{B}|\}, \tag{1}$$

where $b_0 = 0$. The bottom function in the graph on the right is the convex, decreasing, piecewise linear function $l'_{\mathcal{B}}(t)$ defined over the interval $t \in (b_{\theta-1}, b_\theta]$ for all $\theta \in \{1, \ldots, |\mathcal{B}|\}$ as the line through $(b_\theta, h^\mathsf{T} d^*(b_\theta))$ and $(b_{\theta+1}, h^\mathsf{T} d^*(b_{\theta+1}))$. Since $h^\mathsf{T} d^*$ is convex (by Lemma 3.4), we have that $l'_{\mathcal{B}}(t) \leq h^\mathsf{T} d^*(t)$ for all $t \in [0, T)$. Since $h^\mathsf{T} d^*$ is decreasing, we have that $l_{\mathcal{B}}(t) \leq h^\mathsf{T} d^*(t)$ for all $t \in [0, T)$. Thus, the area of each shaded region is a lower bound on the optimal holding cost. Note that, by definition, all three functions $l_{\mathcal{B}}$, $l'_{\mathcal{B}}$, and $h^\mathsf{T} d^*$ concur at the breakpoints in $\mathcal{B}$.

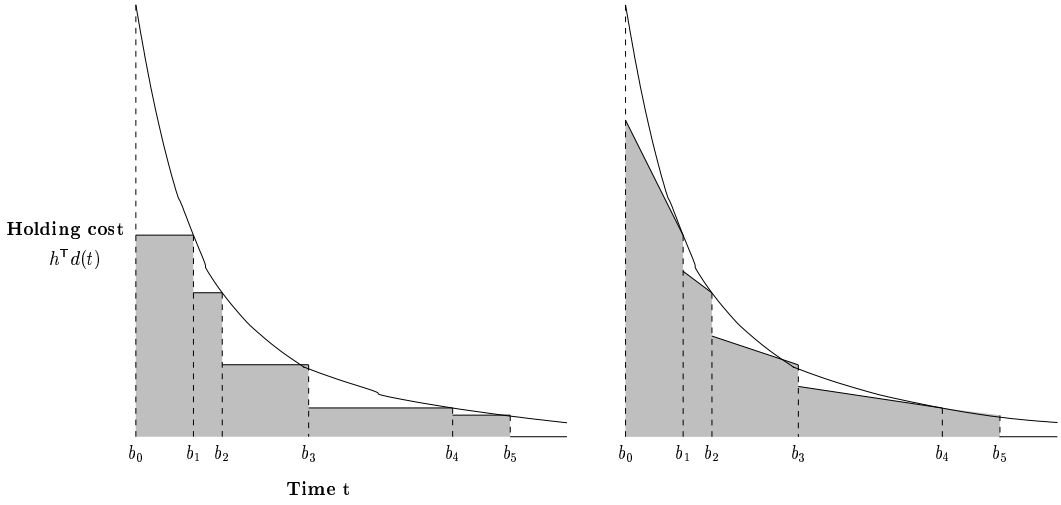

Figure 4: Graphical description of the two lower bounds. The top curve is the function $h^\mathsf{T} d^*$. The shaded region at left shows the area under l. The shaded region at right shows the area under l'.

The area under $l'_{\mathcal{B}}(t)$ in the interval $[b_{\theta-1}, b_\theta)$ is $(b_\theta - b_{\theta-1}) h^\mathsf{T} d^*(b_\theta)$ plus the area of the right triangle above $h^\mathsf{T} d^*(b_\theta)$. This can be computed as one half of the slope of the hypotenuse times the square of the base. This quantity is $\frac{h^\mathsf{T} d^*(b_\theta) - h^\mathsf{T} d^*(b_{\theta+1})}{2(b_{\theta+1} - b_\theta)} (b_\theta - b_{\theta-1})^2$. Defining r_θ as $\frac{b_\theta - b_{\theta-1}}{b_{\theta+1} - b_\theta}$, this whole area can be rewritten as $(b_\theta - b_{\theta-1})\ [(1 + \frac{r_\theta}{2}) h^\mathsf{T} d^*(b_\theta) - \frac{r_\theta}{2} h^\mathsf{T} d^*(b_{\theta+1})]$.

Lemma 3.6 *For any set of positive, ordered breakpoints $\mathcal{B} = \{b_1, \ldots, b_{|\mathcal{B}|} = T\}$,*

$$\int_0^{b_{|\mathcal{B}|}} c^\mathsf{T} f^*(t)\ dt + \sum_{\theta=1}^{|\mathcal{B}|} (b_\theta - b_{\theta-1})\ [(1 + \frac{r_\theta}{2}) h^\mathsf{T} d^*(b_\theta) - \frac{r_\theta}{2} h^\mathsf{T} d^*(b_{\theta+1})] \tag{2}$$

is a lower bound on the cost of an optimal control f^*.

Proof: It suffices to show that $\mathbf{h'} := \sum_{\theta=1}^{|\mathcal{B}|} (b_\theta - b_{\theta-1})\ [(1 + \frac{r_\theta}{2}) h^\mathsf{T} d^*(b_\theta) - \frac{r_\theta}{2} h^\mathsf{T} d^*(b_{\theta+1})]$ is a lower bound on the holding cost of the optimal control. Note that $\mathbf{h'}$ is the integral of

$l'_{\mathcal{B}}(t)$. Since $h^{\mathsf{T}}d^*(t)$ is convex, $l'_{\mathcal{B}}(t) \leq h^{\mathsf{T}}d^*(t)$ for all $t \in (0, b_{|\mathcal{B}|}]$. Thus, $\int_0^{b_{|\mathcal{B}|}} l_{\mathcal{B}'(t)}\ dt \leq \int_0^{b_{|\mathcal{B}|}} h^{\mathsf{T}}d^*(t)\ dt$. □

When $h^{\mathsf{T}}d^*$ is decreasing, we have the following slightly weaker corollary that holds even when $h^{\mathsf{T}}d^*$ is nonconvex.

Corollary 3.7 *For any set of positive, ordered breakpoints $\mathcal{B} = \{b_1, \ldots, b_{|\mathcal{B}|}\}$,*

$$\int_0^{b_{|\mathcal{B}|}} c^{\mathsf{T}} f^*(t)\ dt + \sum_{\theta=1}^{|\mathcal{B}|} (b_\theta - b_{\theta-1})\ h^{\mathsf{T}}d^*(b_\theta) \tag{3}$$

is a lower bound on the cost of an optimal control f^*.

3.3.1 A Computable Lower Bound

Without knowing f^*, we cannot compute the lower bounds described in Lemma 3.6 and Corollary 3.7. In this section, we describe a computable lower bound. This is useful in applying our algorithm in an adaptive setting, as discussed in the remarks at the end of Section 4.1.

We use a slightly different time-expanded network. Let $\mathcal{N}_{\mathcal{B}}^T$ be a condensed time expanded graph with copies of networks corresponding to intervals defined by the breakpoints in the positive, ordered set $\mathcal{B} = \{b_1, \ldots, b_{|\mathcal{B}|}\}$, setting $b_0 = 0$. We omit the use of the intermediate vertices v', by replacing $(v_{b_\theta}, v'_{b_{\theta+1}})$ and $(v'_{b_{\theta+1}}, v_{b_{\theta+1}})$ with $(v_{b_\theta}, v_{b_{\theta+1}})$, and by removing initial arcs (v'_0, v_0). The arc flow costs are as in $\mathcal{N}_{c,\mathcal{B}}^T$.

For the bound corresponding to Lemma 3.6, we compute costs on the holdover arcs as follows. Recall that flow on arc $(v_{b_\theta}, v_{b_{\theta+1}})$ represents the flow at node v at time $b_{\theta+1}$. Thus, the cost on holdover arc $(v_{b_{\theta-1}}, v_{b_\theta})$ is set to $[(1 + \frac{r_\theta}{2})(b_\theta - b_{\theta-1}) - \frac{r_{\theta-1}}{2}(b_{\theta-1} - b_{\theta-2})]h = [(1 + \frac{r_\theta}{2})b_\theta - (1 + \frac{r_\theta + r_{\theta-1}}{2})b_{\theta-1} + \frac{r_{\theta-1}}{2}b_{\theta-2}]h$ for $2 \leq \theta \leq |\mathcal{B}| - 1$, and $(1 + \frac{r_1}{2})(b_1 - b_0)h$ for $\theta = 1$. Denote this network by $\mathcal{N}_{\mathcal{B}}^{\prime T}$.

Lemma 3.8 *If x' is a minimum cost flow in $\mathcal{N}_{\mathcal{B}}^{\prime T}$ for intervals corresponding to breakpoints $\mathcal{B}$, f' is the corresponding control, and d' is the corresponding vector of storage functions, then*

$$\begin{aligned}
&\int_0^{b_{|\mathcal{B}|}} c^{\mathsf{T}} f'(t)\ dt + \sum_{\theta=1}^{|\mathcal{B}|} (b_\theta - b_{\theta-1})\ [(1 + \frac{r_\theta}{2})h^{\mathsf{T}}d'(b_\theta) - \frac{r_\theta}{2}h^{\mathsf{T}}d'(b_{\theta+1})] \\
&= \int_0^{b_{|\mathcal{B}|}} c^{\mathsf{T}} f^*(t) + l'_{\mathcal{B}}(t)dt \\
&\leq \int_0^{b_{|\mathcal{B}|}} c^{\mathsf{T}} f^*(t) + h^{\mathsf{T}}d^*(t)dt.
\end{aligned}$$

Proof: The static flow x' yields a control f' and corresponding storage vector d' that minimizes the quantity $\int_0^{b_{|\mathcal{B}|}} c^\mathsf{T} f'(t)\ dt + \sum_{\theta=1}^{|\mathcal{B}|}(b_\theta - b_{\theta-1})\ [(1+\frac{r_\theta}{2})h^\mathsf{T} d'(b_\theta) - \frac{r_\theta}{2}h^\mathsf{T} d'(b_{\theta+1})]$ over all flows with breakpoints in $\mathcal{B}$. If we average f^* over breakpoints in $\mathcal{B}$, we get a control with cost $\int_0^{b_{|\mathcal{B}|}} c^\mathsf{T} f^*(t)\ dt + \sum_{\theta=1}^{|\mathcal{B}|}(b_\theta - b_{\theta-1})\ [(1+\frac{r_\theta}{2})h^\mathsf{T} d^*(b_\theta) - \frac{r_\theta}{2}h^\mathsf{T} d^*(b_{\theta+1})]$, which is thus an upper bound on the cost of f'. Applying Lemma 3.6 finishes the proof. □

There is also computable bound corresponding to Corollary 3.7 that also uses the same network, but with the cost on holdover arc $(v_{b_\theta}, v_{b_{\theta+1}})$ modified to be $h \times (b_{\theta+1} - b_\theta)$.

4 An Approximation Scheme for Minimum Cost Control

We first describe the approximation scheme for MHC with free flow. In section 4.2, we show how to modify this in the setting of both simple and nonsimple fixed flow paths.

4.1 Free flow controls

Our approximation scheme for MHC uses a time expanded network with network copies representing geometrically increasing units of time. A more complicated time expanded network with geometrically increasing units of time was used previously in [17] for approximating earliest arrival flows with transit times.

Our discretization has size proportional to $\frac{1}{\sqrt{\varepsilon}}$. We begin by presenting a simpler argument that uses a discretization that is linear in $\frac{1}{\varepsilon}$. Our simpler argument uses the lower bound in Corollary 3.7, while the refined argument uses the lower bound in Lemma 3.6. Thus, our simpler argument works even when $h^T d^*$ is nonconvex. We present both proofs, since the first is simpler, while the second is stronger, and builds on the first.

The discretization for the linear-size guarantee uses $\frac{1}{2\varepsilon}(\lceil \log \frac{Th^\mathsf{T} d^\circ \varepsilon}{\delta} \rceil)$ copies of $\mathcal{N}$. These copies are partitioned into $q := \lceil \log \frac{Th^\mathsf{T} d^\circ \varepsilon}{\delta} \rceil$ sets of cardinality $\frac{1}{2\varepsilon}$ each. Denote these sets by $N_0, N_1, \ldots, N_{q-1}$. The size of the intervals in each set depend on a parameter ρ defined as

$$\rho := \delta / h^\mathsf{T} d^\circ.$$

N_0 is the set of intervals of length 2ρ covering interval $[0, \frac{\rho}{\varepsilon})$. N_1 is the set of intervals of length 2ρ covering interval $[\frac{\rho}{\varepsilon}, \frac{2\rho}{\varepsilon})$. For $1 < i < q-1$, N_i is the set of intervals of length $2^i\rho$ covering interval $[\frac{2^{i-1}\rho}{\varepsilon}, \frac{2^i\rho}{\varepsilon})$. N_{q-1} is the set of intervals of length $T\varepsilon$ covering interval $[\frac{T}{2}, T)$. Let $\mathcal{B}'$ be the set of breakpoints corresponding to the endpoints of these intervals.

Theorem 4.1 *A control with cost at most* $(1+\varepsilon)\mathsf{OPT} + \delta$ *and size that is polynomial in* $p(n, m, K, \log U)$, *linear in* $\frac{1}{\varepsilon}$, *and logarithmic in* $\frac{1}{\delta}$ *can be computed in time polynomial in* $p(n, m, K, \log U)$, $\frac{1}{\varepsilon}$, *and* $\log \frac{1}{\delta}$.

Proof: We prove that the control that corresponds to the minimum cost flow x in the time-expanded network based on breakpoints $\mathcal{B}'$ has cost at most $(1+\varepsilon)\mathsf{OPT}+\delta$.

We compare the cost of the control $\bar{f}$ obtained by averaging f^* over each interval defined by consecutive breakpoints in $\mathcal{B}'$ to the lower bound implied by $\bar{f}$ as described in Corollary 3.7. This lower bound is $\int_0^T c^\mathsf{T} f^*(t)\ dt + \int_0^T l_{\mathcal{B}'}(t)\ dt$. Let $\bar{d}$ be the supplies induced by d° and $\bar{f}$. We show that

$$\int_0^T h^\mathsf{T}\bar{d}(t)\ dt \ \leq\ \delta + (1+\varepsilon)\ \int_0^T l_{\mathcal{B}'}(t)\ dt. \tag{4}$$

Since $\bar{f}$ corresponds to a flow in the discretized time expanded network, the control f corresponding to x has cost at most the cost of $\bar{f}$. Combined with the fact that $\int_0^T c^\mathsf{T} f^*(t)\ dt = \int_0^T c^\mathsf{T}\bar{f}(t)\ dt$ and Corollary 3.7, this observation and (4) imply the theorem.

Since $h^\mathsf{T}\bar{d}(t)$ and $l_{\mathcal{B}'}(t)$ are decreasing functions on $(0,T]$, we can evaluate their integrals by considering the area under each curve in horizontal strips.

Consider first the horizontal strip from $h^\mathsf{T}\bar{d}(\frac{\rho}{\varepsilon})$ to $h^\mathsf{T}d^\circ$ as depicted in Figure 5. The area of the difference $h^\mathsf{T}\bar{d}(t) - l_{\mathcal{B}'}(t)$ in this strip can be broken down to the sum of areas of $h^\mathsf{T}\bar{d}(t) - l_{\mathcal{B}'}(t)$ over each interval of length 2ρ. Since $h^\mathsf{T}\bar{d}$ is convex, decreasing, and equals the decreasing step function $l_{\mathcal{B}'}$ at the end points, this difference is the sum of areas of triangles each with base 2ρ, and total height bounded by $h^\mathsf{T}d^\circ$. Thus the difference in the areas in this topmost strip is at most δ.

Now consider any horizontal strip defined by the interval $[\ h^\mathsf{T}\bar{d}(T/2^{j-1}),\ h^\mathsf{T}\bar{d}(T/2^j)]$ for $j = 1,\ldots,q$. We will show that the area under curve $h^\mathsf{T}\bar{d}(t)$ that intersects this strip is at most $1+\varepsilon$ times the area under curve $l_{\mathcal{B}'}(t)$ that intersects this strip. Since this is true for all j; and summed over all j, these strips cover the interval $[0, h^\mathsf{T}\bar{d}(\frac{2\delta}{h^\mathsf{T}d^\circ})]$, this implies inequality (4).

First note that $l_{\mathcal{B}'}(t)$ and $h^\mathsf{T}\bar{d}(t)$ meet at both $t = T/2^j$ and $t = T/2^{j-1}$. Thus, both areas include the area of the strip to the left of $t = \frac{T}{2^j}$: this is the area of the rectangle with height $H_j := h^\mathsf{T}\bar{d}(T/2^j) - h^\mathsf{T}\bar{d}(T/2^{j-1})$ and width $\frac{T}{2^j}$. Both areas include no area to the right of $t = \frac{T}{2^{j-1}}$. Consider now the area in the strip along the horizontal axis from $\frac{T}{2^j}$ to $\frac{T}{2^{j-1}}$. In this interval, time is discretized into intervals of length $\frac{T\varepsilon}{2^j}$. Since $l_{\mathcal{B}'}(t)$ and $h^\mathsf{T}\bar{d}(t)$ agree at all endpoints of these intervals, the area between the $h^\mathsf{T}\bar{d}(t)$ and $l_{\mathcal{B}'}(t)$ in this strip is the area of the triangle with height equal to the height of the strip and base equal to the length of the discretized interval. Thus this area is $H_j \times \frac{T\varepsilon}{2^{j+1}}$. With our previous observations on the area to the left and right in this strip, this implies that in this strip, the ratio of the area under $h^\mathsf{T}\bar{d}(t)$ to the ratio under $l_{\mathcal{B}'}(t)$ is at most $(1+\varepsilon)$. □

Theorem 4.2 *A control with cost at most $(1+\varepsilon)\mathsf{OPT}+\delta$ and size that is polynomial in $p(n,m,K,\log U)$, linear in $\sqrt{\frac{1}{\varepsilon}}$ and logarithmic in $\frac{1}{\delta}$ can be computed in time polynomial in*

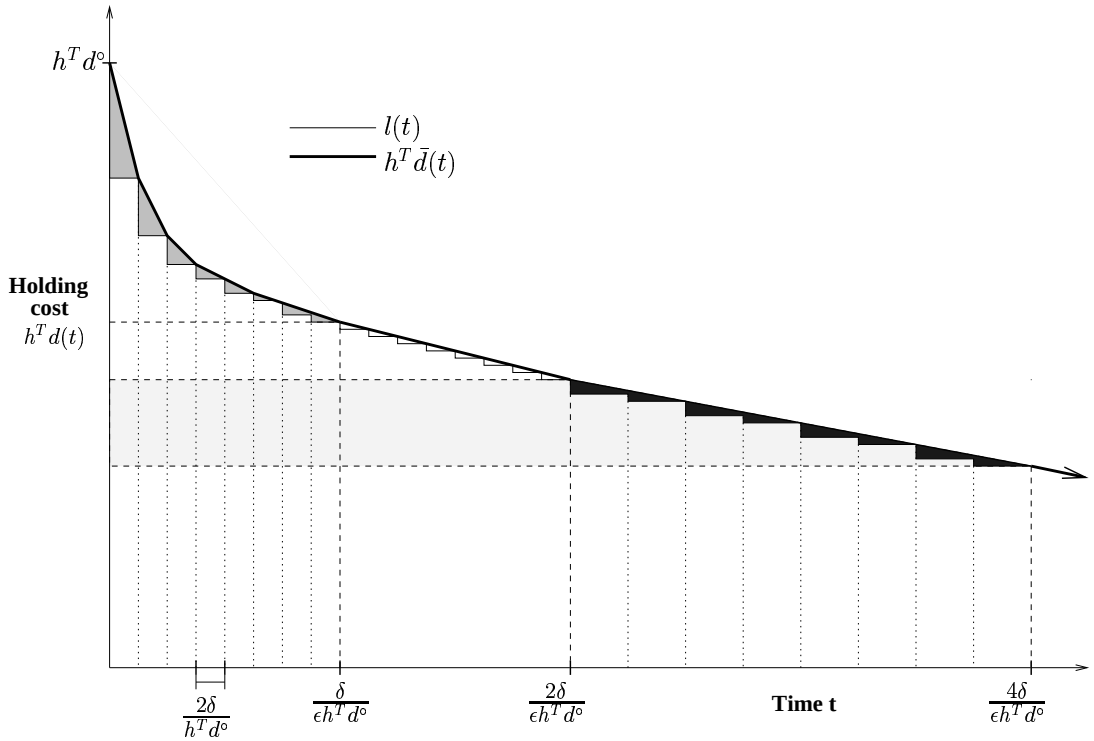

Figure 5: The medium shaded region corresponds to the area of $h^\mathsf{T}\bar{d}(t) - l_{\mathcal{B}'}(t)$ between points $h^\mathsf{T}d^\circ$ and $h^\mathsf{T}\bar{d}(\frac{\delta}{\varepsilon h^\mathsf{T} d^\circ})$ on the vertical axis. The lightly shaded region is the strip for $j = q-2$. The dark shaded region corresponds to the area of $h^\mathsf{T}\bar{d}(t) - l_{\mathcal{B}'}(t)$ between points $h^\mathsf{T}\bar{d}(T/2^{q-3})$ and $h^\mathsf{T}\bar{d}(T/2^{q-2})$ on the vertical axis.

$p(n, m, K, \log U)$, $\frac{1}{\varepsilon}$, *and* $\log \frac{1}{\delta}$.

Proof: We prove that the control that corresponds to the minimum cost flow x in the time-expanded network based on breakpoints $\mathcal{B}'$ has cost at most $(1 + 8\varepsilon^2)\mathsf{OPT} + \delta$.

We compare the cost of the control $\bar{f}$ obtained by averaging f^* over each interval defined by consecutive breakpoints in $\mathcal{B}'$ to the lower bound implied by $\bar{f}$ as described in Lemma 3.6. Let $\bar{d}$ be the supplies induced by d° and $\bar{f}$. This lower bound is the sum of $\int_0^T c^\mathsf{T} f^*(t)\ dt$ and the integral of the convex, decreasing, piecewise linear function $l'_{\mathcal{B}'}(t)$. We show that

$$\int_0^T h^\mathsf{T}\bar{d}(t)\ dt \ \leq\ \delta + (1 + 8\varepsilon^2)\ \int_0^T l'_{\mathcal{B}'}(t)\ dt. \tag{5}$$

Since $\bar{f}$ corresponds to a flow in the discretized time expanded network, the control f corresponding to x has cost at most the cost of $\bar{f}$. Combined with the fact that $\int_0^T c^\mathsf{T} f^*(t)\ dt = \int_0^T c^\mathsf{T}\bar{f}(t)\ dt$ and Lemma 3.6, this observation and (5) imply the theorem.

In the proof of Theorem 4.1, the area between $h^\mathsf{T}\bar{d}(t)$ and $l_{\mathcal{B}'}(t)$ in the interval $[0, \frac{\rho}{\varepsilon})$ is bounded by δ. This argument can be easily extended to hold for the interval $[0, \frac{2\rho}{\varepsilon})$, since the breakpoints in this interval are all the same distance 2ρ apart. Since $l'_{\mathcal{B}'}(t) \geq l_{\mathcal{B}'}(t)$ for all $t \in [0, T]$, the area between $h^\mathsf{T}\bar{d}(t)$ and $l'_{\mathcal{B}'}(t)$ in the interval $[0, \frac{2\rho}{\varepsilon})$ is bounded by δ.

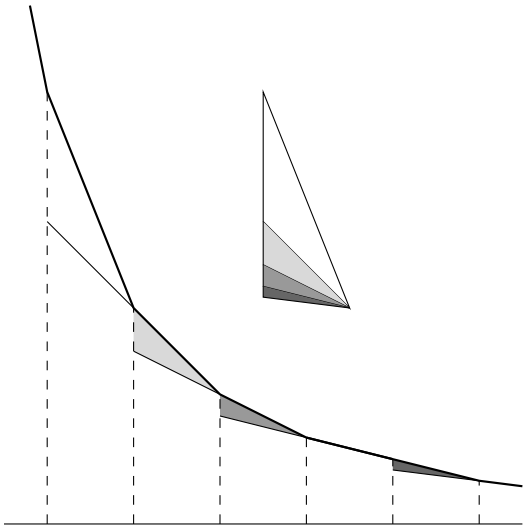

Figure 6: The triangles between $h^\mathsf{T}\bar{d}$, l', and evenly spaced breakpoints stack together to form one triangle that fits inside the triangle between $h^\mathsf{T}\bar{d}$, l, and the first two breakpoints.

We now consider the area between $h^\mathsf{T}\bar{d}(t)$ and $l'_{\mathcal{B}'}(t)$ in the interval $[\frac{2^{j-1}\rho}{\varepsilon}, \frac{2^j\rho}{\varepsilon})$ for $j \geq 2$. This region is divided into $\frac{1}{2\varepsilon}$ triangles: let τ_{ij} be the triangle defined by $l'_{\mathcal{B}'}(t)$, $h^\mathsf{T}\bar{d}(t)$ and the interval $[\frac{2^{j-1}\rho}{\varepsilon} + 2^j\rho(i-1), \frac{2^{j-1}\rho}{\varepsilon} + 2^j\rho i]$, for $i = 1, \ldots, \frac{1}{2\varepsilon}$. See Figure 6. Let σ_{ij} be the corresponding triangle defined by the same interval, $h^\mathsf{T}\bar{d}(t)$, and $l_{\mathcal{B}'}(t)$. Thus $\tau_{ij} \subseteq \sigma_{ij}$. The base of both τ_{ij} and σ_{ij} is $2^j\rho$. The slope of the bottom line of τ_{ij} (i.e the slope of $l'_{\mathcal{B}'}(t)$ in interval $[\frac{2^{j-1}\rho}{\varepsilon} + 2^j\rho(i-1), \frac{2^{j-1}\rho}{\varepsilon} + 2^j\rho i]$) is the same as the slope of the top line of $\tau_{i+1,j}$ (i.e. the slope of $h^\mathsf{T}\bar{d}(t)$ in the interval $[\frac{2^{j-1}\rho}{\varepsilon} + 2^j\rho i, \frac{2^{j-1}\rho}{\varepsilon} + 2^j\rho(i+1)]$). Thus, we can stack these triangles together to form a new triangle of base $2^j\rho$ and height at most $h^\mathsf{T}\bar{d}(\frac{2^{j-1}\rho}{\varepsilon}) - h^\mathsf{T}\bar{d}(\frac{2^j\rho}{\varepsilon})$. This triangle is contained in the triangle σ_{1j}. Thus $\sum_{j=2}^{\lfloor \log \frac{T\varepsilon}{\rho} \rfloor} \mathsf{area}(\sigma_{1j})$ is an upper bound on the area between $h^\mathsf{T}\bar{d}(t)$ and $l'_{\mathcal{B}'}(t)$ in the interval $[\frac{2\rho}{\varepsilon}, T]$. We now bound this area.

Since $h^\mathsf{T}\bar{d}(t)$ is convex and decreasing, we have that

$$h^\mathsf{T}\bar{d}(2^{j-1}\frac{\rho}{\varepsilon} + 2^j\rho(i-1)) - h^\mathsf{T}\bar{d}(2^{j-1}\frac{\rho}{\varepsilon} + 2^j\rho i) \geq h^\mathsf{T}\bar{d}(2^{j-1}\frac{\rho}{\varepsilon} + 2^j\rho i) - h^\mathsf{T}\bar{d}(2^{j-1}\frac{\rho}{\varepsilon} + 2^j\rho(i+1)),$$

for all $j \geq 2$, $1 \leq i \leq \frac{1}{2\varepsilon}$. Thus $\mathsf{area}(\sigma_{ij}) \geq \mathsf{area}(\sigma_{i+1,j})$. Also,

$$h^\mathsf{T}\bar{d}(2^{j-1}\frac{\rho}{\varepsilon} - 2^{j-1}\rho)] - h^\mathsf{T}\bar{d}(2^{j-1}\frac{\rho}{\varepsilon}) \geq \frac{1}{2}[h^\mathsf{T}\bar{d}(2^{j-1}\frac{\rho}{\varepsilon}) - h^\mathsf{T}\bar{d}(2^{j-1}\frac{\rho}{\varepsilon} + 2^j\rho)].$$

for all $j \geq 2$. Thus, $\mathsf{area}(\sigma_{1j}) \leq 4\,\mathsf{area}(\sigma_{i,j-1})$. This implies that $\mathsf{area}(\sigma_{1j}) \leq 8\varepsilon \sum_{i=1}^{1/2\varepsilon} \mathsf{area}(\sigma_{i,j-1})$, and hence $\sum_{j=2}^{\lfloor \log \frac{T\varepsilon}{\rho} \rfloor} \mathsf{area}(\sigma_{1j}) \leq 8\varepsilon \sum_{j=1}^{\lfloor \log \frac{T\varepsilon}{\rho} \rfloor} \sum_{i=1}^{1/2\varepsilon} \mathsf{area}(\sigma_{ij})$. In Theorem 4.1 it is proved that this latter summation is at most $\varepsilon\ \mathsf{OPT}$. Thus we have that $\sum_{j=2}^{\lfloor \log \frac{T\varepsilon}{\rho} \rfloor} \mathsf{area}(\sigma_{1j}) \leq 8\varepsilon^2\ \mathsf{OPT}$, which implies the theorem. □

Remarks 1. The constant in Theorem 4.2 is not tight. With a more careful comparison of areas, it can be shown that the discretization yields a solution of value at most $(1+2\varepsilon^2)\mathsf{OPT}+\delta$.

2. While Theorem 4.2 yields a firm guarantee on the quality of the solution obtained, Lemma 3.8 may be used to obtain a specific guarantee for each particular instance. The specific guarantee may show that the actual approximation is of better quality than Theorem 4.2 promises. Thus, Lemma 3.8 in conjunction with Theorem 3.3 can be used in an iterative manner to find a good discretization for any specific instance: starting with a very coarse discretization, one could iteratively refine only those intervals with large difference between the upper and lower bounds, while leaving large areas of the discretization at a coarse level.

3. In practice, it is desirable to have a control with few breakpoints. Thus, after computing the approximate flow, we can use Lemma 3.8 to remove breakpoints that are not necessary for the approximation guarantee.

4. Theorems 4.1 and 4.2 also hold in the setting of convex flow costs c, since averaging c over an interval only reduces total costs.

4.2 Fixed flow paths

In this section we show how to modify the approach described in the previous sections to handle versions of the problem where the flow path for a commodity is fixed a priori.

Simple paths. If the supply originating at vertex v must follow a fixed path to the sink, we can incorporate this into the time-expanded graph by treating the supply from this sink as a different commodity. In the case when the path is simple, we can force it to follow the path by changing the capacity of arcs not on this path to 0 for this commodity. The resulting problem in the modified time-expanded graph is a multicommodity flow problem on a polynomially sized network, which can be solved in polynomial time via linear programming.

Nonsimple paths. In the case when the path is not simple, we handle the path specification more carefully. In this case, it is not sufficient to restrict the flow of the commodity to arcs on the path, since the flow could then "skip" the cycle, or travel the cycle more times than specified. Instead, we could list the paths in the time-expanded network that the flow

could follow. There are an exponential number of such paths, however, so we cannot afford to list them all explicitly. We argue here that the resulting, path-based linear program can be solved in polynomial time by keeping only an implicit representation of the paths.

We start by describing the path-based linear program corresponding to the time-expanded network with breakpoint set $\mathcal{B}$. Let $\mathcal{P}_k$ be the set of permissible paths for commodity k. For a vector, such as c, defined on the arcs in the time expanded network, we let $c(P) := \sum_{e_\theta \in P} c(e_\theta)$.

minimize

$$\sum_{P \in \mathcal{P}_k} c(P)x(P)$$

subject to

$$\begin{aligned} \sum_{P \in \mathcal{P}_k} x(P) &\geq d_k, \quad k = 1, \ldots, K \\ \sum_k \sum_{P \in \mathcal{P}_k : e_\theta \in P} x(P)/\mu_k(e) &\leq 1, \quad \forall\, e \in A,\ \forall\, \theta \in \mathcal{B}\backslash\{T\}. \end{aligned}$$

This LP has an exponential number of variables. The column pricing problem is, given vectors $w \in \mathbb{R}^{(|\mathcal{B}|-1)\times A}$, find for each commodity k, the permissible path $P \in \mathcal{P}_k$ minimizing

$$c(P) + \sum_{e_\theta \in P} \frac{w_{e_\theta}}{\mu_e}.$$

We can define the distance of edge e for commodity k as $c(e) + w_{e_\theta}/\mu_k(e)$, reducing the pricing problem to a restricted shortest path problem: find the shortest path among all those in $\mathcal{P}_k$. This shortest path problem can be solved exactly by a simple labeling algorithm even if the permissible path for commodity k is non-simple. Fix a commodity k; suppose its associated path visits a node v multiple times, say l times. Then the label for each copy v_θ of v in the time expanded network will be an l tuple $(\gamma_1, \gamma_2, \ldots, \gamma_l)$, with γ_i representing the shortest path from the source to v_θ with i visits to v (including the last). The entry γ_i for node v_θ depends only γ_i for node $v_{\theta-1}$ and the label of its predecessor in this path, and so can be computed efficiently. This labeling scheme can be used to identify the shortest path $P \in \mathcal{P}_k$, solving the pricing problem. This implies, via the ellipsoid algorithm [20], that we can solve the LP in polynomial time.

In practice, we would embed the polynomial time, restricted shortest path subroutine within a column-generation framework for solving these linear programs.

4.3 Infinite capacity arcs

In addition to the modifications suggested at the end of section 4.1, we suggest a modification here that will improve the number of discretizations needed in the case that there are infinite

capacity arcs. In particular, we show how to improve the estimate of the cost computed in the first moments of time in such a case. This is not covered in general by Corollary 3.2, since one simple usefulness of infinite capacity arcs is to allow an arbitrary amount of flow to be transported instantaneously from one node to another. Any flow using infinite capacity arcs in such a manner will not be constant over any non-zero interval of time in which they are used. This is particularly important in the first interval of time. To capture the usage of infinite capacity arcs at time 0, we modify $\mathcal{N}_c^T$ by adding the infinite capacity arcs of $\mathcal{N}$ to the vertex set $V_0' := \{v_0' \mid v \in V \cup \{s\}\}$. That is, for each arc $e \in A$ that has infinite capacity, we include a copy e_0' in V_0' with infinite capacity and the original arc cost. This modified network now allows for instantaneous shipment of flow along infinite capacity arcs at the start of an otherwise piecewise constant control f.

4.4 Continuous input streams

Suppose, in addition to initial storage rates d°, there are also constant rate input vector s that denotes for each node v, the per-unit-time flow arriving at v from outside the network. The new problem becomes one of draining the network stores at minimum holding cost, or reaching the steady state flow of flow in equals flow out at minimum holding cost.

These continuous, constant rate input streams can easily be included in the model as follows. First, ignore the initial storage rates d°, and solve the steady-state problem of sending the incoming flows through the network at minimum flow cost. This induces a residual flow network. The problem of reducing initial stores d° can now be solved in the time expanded graph of this residual network.

4.5 Buffer loss

In some settings with finite buffer capacity, flow may be lost due to buffer overflow. There is a natural penalty for loss of such flow. This can be modelled by introducing an additional node to the time-expanded network to model lost flow, and adding an arc from every vertex to this node with cost equal to the cost of flow loss for that commodity at that node.

5 Generalizations

We consider three distinct generalizations of the basic multiflow problem with holding costs: (a) piecewise constant data; (b) holding cost functions that are convex in the amount of storage; and (c) general constraint matrices. Our treatment so far has been restricted to the case of *constant* data, *linear* holding costs, and a *network* matrix. Each of these generalizations

is fairly straightforward, and is discussed in isolation; however, the algorithm we propose is able to handle any combination of these generalizations.

5.1 Piecewise constant data

Let $\mathcal{J} = \{\beta_0, \beta_1, \ldots, \beta_\rho\}$ be the set of breakpoints of the input data. In the interval $[\beta_i, \beta_{i+1})$, the capacities and costs are constant, while these quantities may change at the points in $\mathcal{J}$. In addition, at each breakpoint $\beta \in \mathcal{J}$ a new set of demands given by vector $d^\circ(\beta)$ may enter the network. Let OPT_ρ be the cost of the optimal solution to this problem.

For this problem, Lemma 3.4 does not hold: it is not true that $h^T d^*$ is a decreasing function, even within an interval $[\beta_i, \beta_{i+1})$.[3] Even though $h^T d^*$ is not decreasing, the next Lemma establishes that it is convex inside each interval $[\beta_i, \beta_{i+1})$, $i = 0, \ldots, \rho - 1$.

Lemma 5.1 *For MHC with piecewise constant data changing at breakpoints in $\mathcal{J}$, the holding cost $h^\mathsf{T} d^*(t)$ is a convex function of t between consecutive breakpoints.*

Proof: The proof is identical to the proof of convexity in Lemma 3.4: since capacity constraints stay constant between breakpoints, averaging the flow between any two points in this interval is feasible and linearizes $h^T d$ between the points. □

We modify the approximation algorithm in Section 4 by modifying the discretization to address the differences between the problem with constant data and the problem with piecewise constant data. Firstly, in the piecewise constant data setting, the initial instantaneous holding cost $h^\mathsf{T} d^\circ$ is no longer a bound on the maximum holding cost. Instead, let $D^\circ := \sum_{\beta \in \mathcal{J}} \sum_v d_v^\circ(\beta)$, and let $H := \max_\beta \max_{v:h_v(\beta)<\infty} h_v(\beta)$. Then $H \times D^\circ$ is an upper bound on the instantaneous holding cost.

More seriously, with piecewise-constant data the expression (3) is no longer a lower bound on the cost of the optimal solution, since $h^\mathsf{T} d^*$ is no longer a decreasing function. Instead, we have the following generalization. For a set $\mathcal{B}$ of breakpoints of the discretization with $\mathcal{J} \subseteq \mathcal{B}$, let $\mathcal{B}^i := \{\beta_i = b_0^i, \ldots, b_{r_i}^i = \beta_{i+1}\}$ be the intersection of $\mathcal{B}$ with $[\beta_i, \beta_{i+1}]$. If the point μ_i that minimizes $h^T d^*$ over the interval $[\beta_i, \beta_{i+1}]$ is in $\mathcal{B}$, then it is easy to see that

$$\int_{\beta_i}^{\beta_{i+1}} c^\mathsf{T} f^*(t)\ dt + \sum_{\theta=1}^{r_i} (b_\theta^i - b_{\theta-1}^i)\ \min\{h^\mathsf{T} d^*(b_\theta^i),\ h^\mathsf{T} d^*(b_{\theta-1}^i)\} \tag{6}$$

is a lower bound on the holding cost of the optimal solution in the interval $[\beta_i, \beta_{i+1})$. If $\mu_i \notin \mathcal{B}$, the next lemma shows that it is still possible to lower bound the cost of the optimal

[3] Consider the network consisting of three nodes $\{a, b, s\}$. There is holding cost 1 at a, holding cost 10 at b, and holding cost 0 at s. There is one unit at a. In the first time unit, there is an arc (a, b) with capacity 1. In the second time unit, there is an arc (b, s) with capacity 1. The instantaneous holding cost starts at 1 and increases linearly to 10 by time 1.

solution with the integral of a function that is constant on intervals of the discretization, for an appropriately defined discretization.

Lemma 5.2 *Let $\mathcal{J} = \{\beta_0, \beta_1, \ldots, \beta_\rho\}$ be the set of breakpoints of the input data to MHC. And let $\mathcal{B}$ with $\mathcal{J} \subseteq \mathcal{B}$ satisfy $\forall i \in \{0, \ldots, \rho-1\}$, that each interval described by consecutive breakpoints in $\mathcal{B}^i$ is adjacent to another such interval of the same or greater length. Then,*

$$\int_0^T c^\mathsf{T} f^*(t)\ dt + \sum_{i=0}^{\rho-1} \sum_{\theta=1}^{r^i} (b^i_\theta - b^i_{\theta-1}) \min\{h^\mathsf{T} d^*(b^i_\theta),\ h^\mathsf{T} d^*(b^i_{\theta-1})\} \tag{7}$$

is a lower bound on the cost of the optimal control.

Proof: As with Corollary 3.7, it suffices to show that $\sum_{i=0}^{\rho-1} \sum_{\theta=1}^{r^i} (b^i_\theta - b^i_{\theta-1}) \min\{h^\mathsf{T} d^*(b^i_\theta),\ h^\mathsf{T} d^*(b^i_{\theta-1})\}$ is a lower bound on the holding cost of the optimal control. On domain $[\beta_i, \beta_{i+1}]$, Lemma 5.1 asserts that $h^\mathsf{T} d^*$ is a convex function. Thus, on domain $[b^i_{\theta-1}, b^i_\theta)$, the function $h^\mathsf{T} d^*$ is either increasing, decreasing, or convex with a minimum in $(b^i_{\theta-1}, b^i_\theta)$. In the first two cases, $(b^i_\theta - b^i_{\theta-1}) \min\{h^\mathsf{T} d^*(b^i_\theta),\ h^\mathsf{T} d^*(b^i_{\theta-1})\}$ is clearly a lower bound on $\int_{b^i_{\theta-1}}^{b^i} h^\mathsf{T} d^*(t)\ dt$. The last case — $h^\mathsf{T} d^*$ is convex with a minimum in $(b^i_{\theta-1}, b^i_\theta)$ — occurs at most once in $[\beta_i, \beta_{i+1}]$. In this case, it could be that the line $\ell(t) = \min\{h^\mathsf{T} d^*(b^i_\theta),\ h^\mathsf{T} d^*(b^i_{\theta-1})\}$ does not satisfy $\ell(t) \le h^\mathsf{T} d^*(t)$ for all $t \in [b^i_{\theta-1}, b^i_\theta)$. (See Figure 7.) We establish the lemma by showing that the area of the region in $[\beta_i, \beta_{i+1}]$ that lies above $h^\mathsf{T} d^*(t)$ but below $\ell(t)$ can be bounded by the area that lies below $h^\mathsf{T} d^*(t)$ but above the lower bound in an interval adjacent to $[b^i_{\theta-1}, b^i_\theta)$.

By the assumptions on $\mathcal{B}$, and without loss of generality, assume that $\theta < r_i$ and $b_\theta - b_{\theta-1} \le b_{\theta+1} - b_\theta$. Specifically, we will argue that the area above $h^T d^*(t)$ but below $\ell'(t) = h^\mathsf{T} d^*(b^i_\theta)$ in $[b^i_{\theta-1}, b^i_\theta)$ is at most the area above $\ell'(t)$ but below $h^T d^*(t)$ in the interval $[b^i_\theta, b^i_{\theta+1})$. Call this first region R and this second region R'. Since the area of R' is slack in the approximation of $h^T d^*$ by $\sum_{\theta=1}^{r^i} (b^i_\theta - b^i_{\theta-1})\ \min\{h^\mathsf{T} d^*(b^i_\theta),\ h^\mathsf{T} d^*(b^i_{\theta-1})\}$, and since $\ell(t) \le \ell'(t)$ for all t, this will establish the lemma.

Since $h^\mathsf{T} d^*$ is convex, there is a subgradient g to $h^\mathsf{T} d^*$ at b_θ that satisfies $g(t) \le h^\mathsf{T} d^*(t)$ for all $t \in [\beta_i, \beta_{i+1}]$. Note that R is completely contained in the triangle τ formed by ℓ' and g in the interval $[b^i_{\theta-1}, b^i_\theta)$. Thus its area is bounded by the area of τ. In contrast, R' completely contains the triangle τ' formed by ℓ and g in the interval $[b_\theta, b_{\theta+1})$. Since τ and τ' are symmetric triangles and τ' is at least as big as τ (by the assumption on θ), we have that the area of R is at most the area of R'. □

Note that if the discretization used in Section 4 contains $\mathcal{J}$, then it satisfies the conditions of Lemma 5.2 as long as there are at least three points of $\mathcal{B}$ in every interval $[\beta_i, \beta_{i+1}]$.

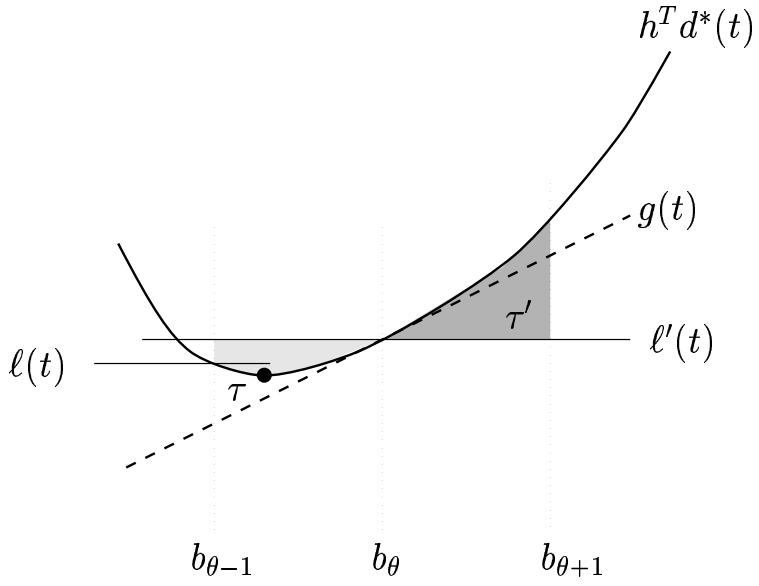

Figure 7: Illustration of the argument in the proof of Lemma 5.2. The lightly shaded region is R, the darker region is R'. The dark point is μ.

In order to obtain a guarantee of $(1+\varepsilon)\mathsf{OPT}_\rho + \delta$ for MHC with piecewise constant data, we will need to use a separate discretization for each interval of the form $[\beta_i, \nu_i]$ or $[\nu_i, \beta_{i+1}]$.

For each interval $[\beta_i, \nu_i]$,, we will make an argument similar to the proof of Theorem 4.1. For each interval $[\nu_i, \beta_{i+1}]$, we use a discretization that starts with small intervals at $\beta_{i+1}]$ and moves to larger intervals as it approaches ν_i. Then we can apply a symmetric argument to that of Theorem 4.1.

Theorem 5.3 *A solution to MHC with piecewise constant cost and capacity functions with at most ρ breakpoints with cost at most $(1+\varepsilon)\mathsf{OPT}_\rho + \delta$ and size linear in ρ, $\frac{1}{\sqrt{\varepsilon}}$, and $\log \delta$, and polynomial in $p(n, m, K, \log U)$ can be computed in time polynomial in $p(n, m, K, \log U)$, ρ, $\frac{1}{\varepsilon}$, and $\log \frac{1}{\delta}$.*

Proof: As with the proof of Theorem 4.1, we compare the cost of the piecewise-constant solution obtained by averaging the optimal solution over intervals in a particular discretization (described below), to the lower bound given by (6) using the same discretization. Due to the similarity with the proof of Theorem 4.1 we describe here only the modifications to the discretization and proof that are necessary to extend that result to piecewise-constant data.

We apply the bounding argument in the proof of Theorem 4.1 to each interval $[\beta_i, \mu_i]$ and $[\mu_i, \beta_{i+1}]$ (one of which may be empty) for all $i = 0, \ldots, \rho - 1$. Since $h^\mathsf{T} d^*$ is decreasing on

$[\beta_i, \mu_i]$ but increasing on $[\mu_i, \beta_{i+1}]$ we use a discretization that is very fine close to β_i and β_{i+1} and course in the middle of the interval $[\beta_i, \beta_{i+1}]$.

The key observations and modifications to the discretization that make this work are as follows:

1. Since μ_i and ν_i won't generally coincide, we will be able to apply the argument in the proof of Theorem 4.1 directly only to one of the two intervals $[\beta_i, \mu_i]$ and $[\mu_i, \beta_{i+1}]$. For the other interval, it is not hard to see that the argument in the proof of Theorem 4.1 is not hurt by consistently replacing sets $\mathcal{N}_j$ for all $j < q'$ with proportionately smaller intervals.

2. To account for the different bound on the maximum instantaneous holding cost, we replace $h^{\mathsf{T}} d^\circ$ with HD° in our determination of the length and number of intervals. Thus, at every breakpoint in $\mathcal{J}$, the discretization starts with intervals of length $\frac{2\delta}{HD^\circ}$.[4]

3. Without further modification, an argument based on this discretization would yield a bound of $(1+\varepsilon)\mathsf{OPT}_\rho + 2\rho\delta$, since we would obtain a δ for the topmost strip in $[\beta_i, \nu_i]$ and another δ for the topmost strip in $[\nu_i, \beta_{i+1}]$, for all $i = 0, \ldots, \rho - 1$. To get an additive factor of just δ, we reduce the length of the initial intervals by dividing them by 2ρ, to get intervals of length $\frac{\delta}{\rho HD^\circ}$. This results in a total additive factor of δ for all the topmost strips in the intervals $[\beta_i, \nu_i]$ and $[\nu_i, \beta_{i+1}]$. To maintain the multiplicative factor of $(1+\varepsilon)$ on all other horizontal strips, we must divide the length of all other intervals also by 2ρ.

Thus, within $[\beta_i, \nu_i]$, we use a discretization with $\frac{1}{2\varepsilon}(\lceil \log \frac{\rho(\beta_{i+1}-\beta_i)HD^\circ}{\delta\varepsilon} \rceil)$ intervals. These intervals are partitioned into $q := \lceil \log \frac{\rho(\beta_{i+1}-\beta_i)HD^\circ}{\delta\varepsilon} \rceil)$ sets of cardinality $\frac{1}{2\varepsilon}$ each. The first two sets have intervals of length $\frac{\delta}{\rho HD^\circ}$. Interval lengths double successively in subsequent sets. The intervals that cover $[\nu_i, \beta_{i+1}]$ start at β_{i+1} at length $\frac{\delta}{\rho HD^\circ}$, and increase as they move back towards ν_i. The total number of breakpoints over $[0, T]$ is thus at most $\frac{\rho}{\varepsilon}(\lceil \log \frac{\rho THD^\circ}{\delta\varepsilon} \rceil)$, and hence the solution complexity is polynomial in $p(n, m, K, \log U)$, $\frac{1}{\varepsilon}$, and $\log \frac{1}{\delta}$. □

5.2 Convex holding costs

Let $h_{k,v}(x)$ be the instantaneous holding cost incurred in storing x units of commodity k at node v. We assume that $h_{k,v}(x)$ is convex in x, for all k and $v \in V$. Let OPT_c be the cost

[4]The use of HD° is a coarse overestimate. This could be reduced in practice (and in theory) by replacing this with the maximum holding cost in this interval, times the sum of demands input up through this interval minus the time elapsed to the start (or end) of the interval.

of the optimal solution. As before, any control f to MHC determines $d_k(v,t)$, the storage of commodity k at node v at time t. Let

$$H^*(t) = \sum_k \sum_{v \in V} h_{k,v}(d_k^*(v,t)),$$

where d_k^* is induced by an optimal control f^*. The extension of Lemma 3.4 to this setting is the following.

Lemma 5.4 *$H^*(t)$ is a convex, decreasing function of t.*

Proof: The proof is similar to the proof of Lemma 3.4. If H^* is not convex, then for some t_1, t_2, $H^*(\lambda t_1 + (1-\lambda)t_2) > \lambda H^*(t_1) + (1-\lambda)H^*(t_2)$ for every $\lambda \in (0,1)$. Modify f on the interval $[t_1, t_2)$ by replacing $f(e,t)$ with the average flow rate $\frac{1}{t_2 - t_1} \int_{t_1}^{t_2} f(e,t)\, dt$ for all $e \in A$ and all $t \in [t_1, t_2)$. Call the new control $\bar{f}$. Since f obeys capacity constraints, so does $\bar{f}$. Note that $d_{\bar{f}}(t_1) = d^*(t_1)$ and $d_{\bar{f}}(t_2) = d^*(t_2)$ but that for $t \in (t_1, t_2)$, $d_{\bar{f}}$ changes linearly from $d^*(t_1)$ to $d^*(t_2)$; i.e. $d_{\bar{f}}(t) = \frac{t_2 - t}{t_2 - t_1} d^*(t_1) + \frac{t - t_1}{t_2 - t_1} d^*(t_2)$. Since d^* is nonnegative, so is $d_{\bar{f}}$. The instantaneous holding cost incurred at any time $t \in [t_1, t_2)$ under control $\bar{f}$ is

$$\sum_k \sum_{v \in V} h_{k,v} \left(\frac{t_2 - t}{t_2 - t_1} d_{k,v}^*(t_1) + \frac{t - t_1}{t_2 - t_1} d_{k,v}^*(t_2) \right),$$

which, by convexity of the storage cost function $h_{k,v}$, is at most

$$\frac{t_2 - t}{t_2 - t_1} \sum_k \sum_{v \in V} h_{k,v}(d_{k,v}^*(t_1)) + \frac{t - t_1}{t_2 - t_1} \sum_k \sum_{v \in V} h_{k,v}(d_{k,v}^*(t_2)).$$

By the definition of H^* and the choice of t_1, t_2, the latter expression is

$$\frac{t_2 - t}{t_2 - t_1} H^*(t_1) + \frac{t - t_1}{t_2 - t_1} H^*(t_2) \;<\; H^*(t),$$

contradicting the optimality of f^*. Hence $H^*(t)$ is convex in t.

As before, since $H^*(0) > 0$, $H^*(T) = 0$, $H^*(t) \geq 0$, and H^* is convex, H^* is also a decreasing function. □

Consider the interval partition $\mathcal{I}$ with breakpoints $0 = b_0 < b_1 < \cdots < b_r = T$. Since H^* is a decreasing function, the expression

$$\int_0^T c^{\mathsf{T}} f^*(t)\; dt + \sum_{\theta=1}^{r} (b_\theta - b_{\theta-1})\; H^*(b_\theta)$$

is a lower bound on the optimal cost.

We now explain briefly why the approximation scheme outlined in Section 4.1 and its analysis remain valid for the case of convex holding costs. For convenience, we reproduce

the description of the time expanded network used there; note that $H(0)$, the instantaneous holding cost incurred at time 0, is independent of the control used. The discretization uses $\frac{1}{2\varepsilon}(\lceil \log \frac{TH(0)}{\delta\varepsilon} \rceil)$ copies of $\mathcal{N}$, partitioned into $q := \lceil \log \frac{TH(0)}{\delta\varepsilon} \rceil$ sets of cardinality $\frac{1}{2\varepsilon}$ each. Denote these sets by $N_0, N_1, \ldots, N_{q-1}$. N_0 is the set of intervals of size $\frac{2\delta}{H(0)}$ covering interval $[0, \frac{\delta}{\varepsilon H(0)})$. N_1 is the set of intervals of size $\frac{2\delta}{H(0)}$ covering interval $[\frac{\delta}{\varepsilon H(0)}, \frac{2\delta}{\varepsilon H(0)})$. For $1 < i < q-1$, N_i is the set of intervals of size $\frac{2^i\delta}{H(0)}$ covering interval $[\frac{2^{i-1}\delta}{\varepsilon H(0)}, \frac{2^i\delta}{\varepsilon H(0)})$. N_{q-1} is the set of intervals of size $\frac{T\varepsilon}{2}$ covering interval $[\frac{T}{2}, T)$. Let $\mathcal{I}'$ be the set of all these intervals, and let B' be the set corresponding to the endpoints of these intervals.

Theorem 5.5 *The control f that corresponds to the minimum cost flow x in the time-expanded network based on intervals $\mathcal{I}'$ has cost at most $(1+\varepsilon)\mathsf{OPT}_c + \delta$.*

Proof: Let $l(t) := H(b_\theta)$ for all $t \in (b_{\theta-1}, b_\theta]$, for all $\theta = 1, \ldots r$. Let $\bar{f}$ be obtained by averaging f^* over each interval in $\mathcal{I}'$, inducing the storage function $\bar{d}$ with the corresponding holding cost function $\bar{H}$. As in the proof of Theorem 4.1, our task essentially reduces to bounding the area between the curves $\bar{H}(t)$ and $l(t)$. Specifically, the theorem follows if we show that

$$\int_0^T \bar{H}(t)\ dt \ \leq\ \delta + (1+\varepsilon)\ \int_0^T l(t)\ dt$$

Since $\bar{H}(t)$ and $l(t)$ are decreasing functions on $(0, T]$, we can evaluate their integrals by considering the area under each curve in horizontal strips. Note that $\bar{H}(t) = l(t)$ for all $t \in B'$. Consider first the horizontal strip from $\bar{H}(\frac{\delta}{\varepsilon H(0)})$ to $H(0)$. The area of the difference $\bar{H}(t) - l(t)$ in this strip is the sum of areas of $\bar{H}(t) - l(t)$ over each interval of size $\frac{2\delta}{H(0)}$. Since $\bar{H}$ is convex, decreasing, and equals the decreasing step function l at the end points, this difference is *at most* the sum of areas of triangles, each with base $\frac{2\delta}{H(0)}$, and total height bounded by $H(0)$; thus, the difference in the areas in this topmost strip is at most δ. A similar argument establishes that the area between the curves $\bar{H}(t)$ and $l(t)$ in any horizontal strip defined by the interval $[\bar{H}(T/2^{j-1}),\ \bar{H}(T/2^j)]$ (for $j = 0, \ldots, q-1$) is within $(1+\varepsilon)$ of the area under $l(t)$.

□

Corollary 5.6 *A solution to the MHC problem with convex holding costs of value at least $(1+\varepsilon)\mathsf{OPT}_c + \delta$ with size linear in $\frac{1}{\sqrt{(\varepsilon)}}$ and $\log \frac{1}{\delta}$ and polynomial in $p(n, m, K, \log U)$ can be computed in time polynomial in $p(n, m, K, \log U)$, $\frac{1}{\varepsilon}$, and $\log \frac{1}{\delta}$.*

5.3 General dynamics

Consider the following problem, which we will call the generalized multiflow problem with holding costs:

$$\text{Minimize} \qquad \int_0^T \{cf(t) + hd(t)\}\, dt$$

subject to

$$\begin{aligned} d(t) &= d(0) - \int_0^t G\, f(s) ds, \quad t \in [0,T] \\ Hf(t) &\leq 1, \quad t \in [0,T] \\ d(t) &\leq a, \quad t \in [0,T] \\ d(t), f(t) &\geq 0, \quad t \in [0,T]. \end{aligned}$$

Here c, h, $f(t)$, $d(t)$, and a are vectors, and G, H are matrices with appropriate dimensions. Let OPT_g be the objective function value of the optimal solution. Our treatment of MHC so far has been restricted to the case in which G is a network matrix. However, there are practical settings in which G is not a network matrix. For instance, consider a flexible service system to which m different types of jobs arrive; the service system may be operated in any one of n *processing configurations*. Each configuration specifies the rate at which the m job types are processed. (Note that configurations may process multiple job types simultaneously.) Suppose we wish to operate this service facility so as to optimize some measure of the jobs in the system. A (crude) model for this problem is to let the rows and columns of G represent the m job types and the n processing configurations respectively; thus, G_{ij} would represent the quantity of job type i processed per unit time while operating the service facility in configuration j. The f vector tracks the amount of time spent in each configuration, and the d vector keeps track of the number of jobs in the system. (The other constraints admit the usual interpretation.) Such models and their extensions have been the subject of recent papers [19, 25], to which we refer the interested reader; we simply note that continuous linear programming problems with general G matrices arise as useful models in connection with complex scheduling problems. We outline briefly how the methods proposed here naturally extend to the case of a general G matrix.

If G is not a network matrix, there is no natural interpretation of the discretization in terms of a time expanded network. Instead, it is convenient to view the discretized problem as a (large) linear programming problem. The discretization remains the same as the one used in Section 4.1, with the breakpoints denoted by $0 = b_0, b_1, \ldots, b_r = T$, where $r = \frac{1}{2\varepsilon}(\lceil \log \frac{TH(0)}{\delta\varepsilon} \rceil)$. The continuous linear programming problem then reduces to solving the

linear programming problem

$$\text{Minimize} \qquad \sum_{\theta=1}^{r} \{cf(\theta) + \frac{(b_\theta - b_{\theta-1})}{2}(hd(\theta) + hd(\theta-1))\}$$

subject to

$$\begin{aligned} d(\theta) &= d(\theta-1) - G\, f(\theta)(b_\theta - b_{\theta-1}), \quad \theta = 1, 2, \ldots, r \\ Hf(\theta) &\leq 1, \quad \theta = 1, 2, \ldots, r \\ d(\theta) &\leq a, \quad \theta = 1, 2, \ldots, r \\ d(\theta), f(\theta) &\geq 0, \quad \theta = 1, 2, \ldots, r. \end{aligned}$$

In this discrete linear program (DLP), the variables $f(\theta)$ represent the constant flow rate in the interval $(b_{\theta-1}, b_\theta]$, and the variables $d(\theta)$ represent the storage at time b_θ; the storage at any intermediate time epoch $t \in (b_{\theta-1}, b_\theta)$ varies linearly between $d(\theta-1)$ and $d(\theta)$.

Clearly, the instantaneous holding cost function induced by an optimal control remains convex and decreasing (analog of Lemma 3.4). This property implies the following result via an argument identical to the one used in proving Theorem 4.1; we omit the details.

Theorem 5.7 *A solution to the generalized MHC problem with cost at most* $(1+\varepsilon)\mathsf{OPT}_g + \delta$ *can be computed in time polynomial in* $p(n, m, K, \log U)$, $\frac{1}{\varepsilon}$, *and* $\log \frac{1}{\delta}$.

General Convex Constraints The instantaneous holding cost function induced by an optimal control also remains convex and decreasing when there are general convex constraints in addition to linear constraints. Thus our approximation guarantees also hold in this context.

6 Conclusions

We have described an algorithm that finds a solution to the multicommodity flow problem with holding costs (MHC) that has value at most $(1+\varepsilon)\mathsf{OPT} + \delta$ and runs in time polynomial in $p(n, m, K, \log U)$, $\frac{1}{\varepsilon}$ and $\log \delta$. The MHC problem is motivated by fluid relaxations of stochastic scheduling problems and has been studied before as a motivating special case of separated continuous linear programming. Our algorithm guarantees simultaneously

- a polynomial bound on the run time,
- a solution with value that is arbitrarily close to the optimal solution, and
- a solution with size that is polynomial in the size of the input.

This last property is especially significant since optimal solutions may have size that is exponential in the size of the input.

Moreover, our algorithm is practical: it requires solving just one polynomially-sized linear program. In addition, we provide a strong lower bound that may be used to obtain good solutions with fewer breakpoints.

Finally, we show that our algorithm is quite general: modifications of it work to provide the same guarantees with convex holding costs, piecewise constant data, and arbitrary convex constraints.

Acknowledgments

We are grateful to the referees for their very careful reading of our initial draft, and their useful feedback. This led to an improved presentation of this paper.

References

[1] E. J. Anderson, *A continuous model for job-shop scheduling*, PhD thesis, University of Cambridge, 1978.

[2] E. J. Anderson and P. Nash, *Linear Programming in Infinite-Dimensional Spaces*, John Wiley & Sons, New York, 1987.

[3] E. J. Anderson, P. Nash, and A. F. Perold, *Some properties of a class of continuous linear programs*, SIAM J. Control and Optimization, 21 (1983), pp. 758–765.

[4] F. Avram, D. Bertsimas, and M. Ricard, *Fluid models of sequencing problems in open queueing networks: an optimal control approach*, in Stochastic Networks, F. P. Kelly and R. J. Williams, eds., vol. 71 of Proceedings of the International Mathematics Association, Springer-Verlag, New York, 1995, pp. 199–234.

[5] F. Avram, D. Bertsimas, and J. Sethuraman, *Optimal control of fluid tandem networks*, Manuscript in preparation, (2002).

[6] N. Bauerle, *Asymptotic optimality of tracking policies in stochastic networks*, Annals of Applied Probability, 10 (2000), pp. 1065–1083.

[7] R. Bellman, *Bottleneck problem and dynamic programming*, Proc. Nat. Acad. Sci., 39 (1953), pp. 947–951.

[8] ——, *Dynamic Programming*, Princeton University Press, New Jersey, 1957.

[9] D. Bertsimas, D. Gamarnik, and J. Sethuraman, *From fluid relaxations to practical algorithms for high multiplicity job shop scheduling: the holding cost objective*, Operations Research, (2002).

[10] D. Bertsimas and J. Sethuraman, *From fluid relaxations to practical algorithms for job shop scheduling: the makespan objective*, Mathematical Programming, 92 (2002), pp. 61–102.

[11] H. Chen and A. Mandelbaum, *Discrete flow networks: Bottleneck analysis and fluid approximations*, Mathematics of Operations Research, 16 (1991), pp. 408–446.

[12] ——, *Hierarchical modeling of stochastic networks, part i: fluid models*, in Stochastic Modeling and Analysis of Manufacturing Systems, D. D. Yao, ed., New York, NY, 1994, Springer-Verlag, pp. 47–105.

[13] H. Chen and D. D. Yao, *Fundamentals of Queueing Networks: Performance, Asymptotics, and Optimization*, Springer-Verlag, New York, 2001.

[14] J. G. Dai and G. Weiss, *A fluid heuristic for minimizing makespan in job-shops*, Operations Research, (2002).

[15] J. Filipiak, *Modelling and Control of Dynamic Flows in Communication Networks*, Springer Verlag, Berlin, 1988.

[16] L. Fleischer, *Faster algorithms for the quickest transshipment problem*, SIAM J. on Optimization, 12 (2001), pp. 18–35.

[17] L. Fleischer and M. Skutella, *The quickest multicommodity flow problem*, in 9th International Integer Programming and Combinatorial Optimization Conference, no. 2337 in LNCS, 2002, pp. 36–53.

[18] ——, *Quickest flows over time.* Submitted, 2003.

[19] N. Gans and G. van Ryzin, *Optimal control of a multiclass, flexible queueing system*, Operations Research, 45 (1997), pp. 677–693.

[20] M. Grötschel, L. Lovász, and A. Schrijver, *The ellipsoid method and its consequences in combinatorial optimization*, Combinatorica, 1 (1981), pp. 169–197.

[21] B. Hajek and R. G. Ogier, *Optimal dynamic routing in communication networks with continuous traffic*, Networks, 14 (1984), pp. 457–487.

[22] J. M. Harrison, *Brownian motion and stochastic flow systems*, John Wiley & Sons, 1985.

[23] ——, *Brownian models of queueing networks with heterogenous customer populations*, in Stochastic Differential Systems, Stochastic Control Theory and Applications, W. Fleming and P. L. Lions, eds., Proceedings of the International Mathematics Association, Springer-Verlag, 1988, pp. 147–186.

[24] ——, *The bigstep approach to flow management in stochastic processing networks*, in Stochastic Networks: Theory and Applications, F. P. Kelly, S. Zachary, and I. Ziedins, eds., Oxford University Press, 1996, pp. 57–90.

[25] ——, *Stochastic networks and activity analysis*, in Analytic Methods in Applied Probability. In Memory of Fridrih Karpelevich, Y. Suhov, ed., American Mathematical Society, 2002.

[26] B. Hoppe and É. Tardos, *Polynomial time algorithms for some evacuation problems*, in Proc. of 5th Annual ACM-SIAM Symp. on Discrete Algorithms, 1994, pp. 433–441.

[27] X. Luo and D. Bertsimas, *A new algorithm for state-constrained separated continuous linear programs*, SIAM Journal on control and optimization, 37 (1999), pp. 177–210.

[28] C. Maglaras, *Dynamic Control of Stochastic Processing Networks: A Fluid Model Approach*, PhD thesis, Stanford University, August 1998.

[29] ——, *Discrete-review policies for scheduling stochastic networks: Trajectory tracking and fluid-scale asymptotic optimality*, Annals of Applied Probability, 10 (2000), pp. 897–929.

[30] S. P. Meyn, *Stability and optimization of queueing networks and their fluid models*, in Mathematics of Stochastic Manufacturing Systems, G. G. Yin and Q. Zhang, eds., vol. 33 of Lectures in Applied Mathematics, American Mathematical Society, 1997, pp. 175–200.

[31] A. B. Philpott and M. Craddock, *An adaptive discretization algorithm for a class of continuous network programs*, Networks, 26 (1995), pp. 1–11.

[32] M. C. Pullan, *An algorithm for a class of continuous linear programs*, SIAM Journal on Control and Optimization, 31 (1993), pp. 1558–1577.

[33] ——, *On the solution of a class of continuous linear programs*, SIAM Journal on Control and Optimization, 32 (1994), pp. 1289–1296.

[34] ——, *Forms of optimal solutions for separated continuous linear programs*, SIAM Journal on Control and Optimization, 33 (1995), pp. 1952–1977.

[35] ——, *A duality theory for separated continuous linear programs*, SIAM Journal on Control and Optimization, 34 (1996), pp. 931–965.

[36] G. Rote. Personal communication, 2002.

[37] J. Sethuraman, *Scheduling Multiclass Queueing Networks and Job Shops using Fluid and Semidefinite Relaxations*, PhD thesis, Massachusetts Institute of Technology, September 1999.

[38] G. Weiss, *A simplex based algorithm to solve separated continuous linear programs.* Unpublished manuscript, 2002.